UNIVERSIDAD CARLOS III DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

TRABAJO DE FIN DE GRADO

Simulador de EGSE (Electric Ground Support Equipment) [TELEMETRÍA] mediante plataforma Software Defined Radio

Departamento de Teoría de la Señal y Comunicaciones
Grado en Ingeniería en Tecnologías de Telecomunicación

Autor: FÉLIX JIMÉNEZ MONZÓN

Tutor: DR. VÍCTOR P. GIL JIMÉNEZ

Leganés, septiembre de 2015

Toda tecnología lo suficientemente avanzada
es indistinguible de la magia.

Arthur C. Clarke
(1917 - 2008)

Agradecimientos

Quiero agradecer en primer lugar al Dr. Víctor P. Gil Jiménez por haber pensado en mí a la hora de hacer este proyecto y por haberme dado la oportunidad de llevarlo a cabo y de trabajar junto a él.

En segundo lugar, a mis compañeros de proyecto Dani y Fernando, ya que trabajar a su lado ha sido una experiencia enriquecedora que espero que se repita en el futuro.

En tercer lugar, a mis amigos de la Universidad, en especial a Pablo, Javi, Escudero, Paloma y Patri, que han hecho que estos años en la Universidad sean unos de los mejores de mi vida. Y también a mis amigos de toda la vida, sobre todo a Mara y Xavi, que independientemente de las circunstancias siempre han estado ahí.

Y por último, y no por ello menos importantes, quiero dar también las gracias a mis padres y a mi familia. A mis padres, porque sin ellos habría sido imposible llegar hasta aquí y ser quien soy. Y al resto de mi familia, por haberme apoyado siempre y creer en mí.

Me gustaría hacer también una especial mención a mi tío Javi, que hace poco nos dejó, y a quien nunca olvidaré.

A todos vosotros. Gracias.

Resumen

En este proyecto se ha realizado la implementación de un simulador de una comunicación por satélite. Es un proyecto conjunto que ha sido llevado a cabo por tres alumnos de la Universidad Carlos III de Madrid. Cada uno de estos alumnos se ha encargado de diseñar e implementar una parte de la comunicación por satélite.

El objetivo de este Trabajo de Fin de Grado es realizar un simulador de EGSE (Electric Ground Support Equipment), es decir, de una estación de tierra en una comunicación satélite-base. Concretamente, en este proyecto se ha llevado a cabo la implementación de las funciones de Telemetría de la estación de tierra. Se ha realizado utilizando una plataforma SDR (Software Defined Radio), que es una tecnología que permite implementar sistemas de radio en software. Y para ello se ha empleado un transceptor NI USRP (Universal Software Radio Peripheral de National Instruments®).

La programación de los transceptores NI USRP se ha llevado a cabo mediante la plataforma LabVIEW, un lenguaje y a la vez entorno de programación gráfica basada en flujo de datos que permite crear sistemas de comunicaciones inalámbricas.

El objetivo del proyecto es, por tanto, interpretar los datos de telemetría enviados por un satélite y actuar en consecuencia, logrando así la simulación de una comunicación satélite-tierra con éxito.

Índice

Índice de figuras

Índice de tablas

Glosario

ACK	Acknowledgement	Asentimiento
ADC	Analog to Digital Converter	Conversor Analógico-Digital
APID	Application Process ID	Identificador del Proceso de Aplicación
BPSK	Binary Phase-Shift Keying	
CNAF		Cuadro Nacional de Atribución de Frecuencias
DAC	Digital to Analog Converter	Conversor Digital-Analógico
DDC	Digital DownConversion	Conversión Digital hacia abajo
DUC	Digital UpConversion	Conversión Digital hacia arriba
ECSS	European Cooperation for Space Standardization	Cooperación Europea para la Estandarización Espacial
EGSE	Electric Ground Support Equipment	Equipo Eléctrico de Apoyo en Tierra
HK	Housekeeping	Gestión interna
ID	Identifier	Identificador
IVA		Impuesto sobre el Valor Añadido
LabVIEW	Laboratory Virtual Instrument Engineering Workbench	
NI	National Instruments®	
PEC	Packet Error Control	Control de Errores en Paquetes
PERT	Project Evaluation and Review Techniques	Técnicas de Revisión y Evaluación de Proyectos

PN	Pseudo-Noise	Pseudoruido/Pseudoaleatorio
PUS	Packet Utilization Standard	Estándar de la Utilización de Paquetes
RAM	Random Access Memory	Memoria de acceso aleatorio
RID	Report Identification	Identificador de reporte
ROM	Read Only Memory	Memoria de Solo Lectura
RX	Reception	Recepción
SAR	Specific energy Absorption Rate	Índice de Absorción Específica de Energía
SDR	Software Defined Radio	Radio definido por software
SID	Structure Identification	Identificador de estructura
TC	Telecommand	Telecomando
TFG		Trabajo de Fin de Grado
TM	Telemetry	Telemetría
TX	Transmission	Transmisión
USRP	Universal Software Radio Peripheral	Periférico software universal para radio
VI	Virtual Instrument	Instrumento Virtual

Extended abstract

Nowadays, telecommunications have reached a fundamental importance in our society. A developed society cannot be understood without telecommunications. They have an essential role in our daily lives because they permit to communicate any place of the Earth with any other in an almost instantaneous period of time.

There are several fields of telecommunications, but this project is focused on satellite communications [1]. This type of communications is also indispensable because its applications are almost infinite. They are used both in mobile communications and in space exploration, among others. In this project, a simulator of a satellite communication is developed.

This project has been carried out by three students from the Carlos III University of Madrid. Each student has designed and developed one part of the satellite communication and, when the three parts are joined together, a complete satellite communication system can be simulated.

The three students who have performed this project are: Daniel Rodríguez Mogollón, which Bachelor Thesis can be found in reference [2]; Fernando García Arias, which Bachelor Thesis can be found in reference [3]; and Félix Jiménez Monzón, who has carried out the current Bachelor Thesis.

The idea of the project was proposed by the tutor, Dr. Víctor P. Gil Jiménez. It arose due to the lack of a simulator of a satellite communication system in the Carlos III University of Madrid. This project has been made with the objective of helping future students to learn more easily how a satellite communication [1] works. Students would understand better these subjects with a demonstration of its operation than with a theoretical explanation.

The objectives of this project were also fixed at the beginning thereof. There were three main objectives:

Firstly, design and develop a simulation of a complete satellite communication [1]. This is a common objective for the three members of the project team.

Secondly, implement successfully the telemetry functionalities of a ground station which belong to a satellite communication. This objective only affects this Bachelor Thesis.

And in third place, carry out a program as easy to operate and as easy to understand as possible. This is because if someday someone wants to use it or to improve it, the program has to be easy to comprehend.

Besides these, there are other objectives posed at the beginning of the project. These include, firstly, deepen into the subjects studied during the degree, especially into the satellite related subjects; secondly, investigate about the technology used in this project, which was almost unknown for the project team; and thirdly, as a result of the previous two, increase knowledge on the issues covered in the project.

In this project, the scenario of the satellite communication which is going to be simulated could be as the following:

Figure 0.1: Satellite communication [Figure 1.1 [1]]

As seen in the picture above, in a satellite communication two main elements intervene. These are the satellite and the ground station.

The satellite is only a kind of a repeater station that receives signals from Earth, processes them and retransmits them back to the Earth. The objective of the satellite (in this project it has been simulated an observation satellite) is to take data about some parameters, to perform the tasks ordered from the ground station, and to send the information back to the ground station.

On the other hand, the ground station is a terminal located on the surface of the Earth that communicates with the satellite. The objective of the ground station is to control the satellite and to receive and interpret the information sent from the satellite.

All the ground stations have a transmission and/or reception system. The transmission system is carried out by implementing telecommand commands, while the receiving system is carried out by implementing telemetry commands.

In this Bachelor Thesis, it has been designed and carried out therefore a simulator of the telemetry functions of a ground station EGSE (Electric Ground Support Equipment) that belongs to a satellite communications system.

The project has been developed in a software platform using SDR (Software Defined Radio) technology [4] [5]. This technology is, increasingly, more important when developing new technologies. SDR is simply defined like: “Radio in which some or all of the physical layer functions are software defined” [6].

There are a lot of advantages of using the SDR technology, but the main one is the flexibility, because it permits to change the topology of the radio systems without changing the hardware. This allows the radio services providers to develop and introduce rapidly and easily new services, and reducing notably the price of making changes in the network.

This SDR technology [4] [5] [6] has been essential in this project because everything has been developed in a SDR platform. This has been done using a NI USRP (Universal Software Radio Peripheral of National Instruments®) transceiver.

A NI USRP [7] is a transceiver which can transmit and receive radio signals within a specific bandwidth. Due to its great capacity for flexibility and adaptation, they are widely used both in educational areas and in research areas such as radiofrequency. To be precise, in this project it has been used the NI USRP 2920 model [8] [9] [10]. It can be seen in the picture below.

Figure 0.2: NI USRP 2920 transceiver [8]

They can work in a frequency range from 50 MHz to 2.2 GHz. In this project, the transmission and receiving frequencies used are: 650 MHz for the uplink (ground station -> satellite), and 600 MHz for the downlink (satellite -> ground station). These frequencies have been used to avoid the interferences problem because they belong to a frequency range rarely used and reserved for television [11] [12].

These NI USRP transceivers have been programmed using a programming environment also developed by National Instruments® [13]. This programming environment is LabVIEW.

LabVIEW (Laboratory Virtual Instrument Engineering Workbench) [14] [15] [16] is a programming language and also a graphical programming environment. LabVIEW uses a programming language, also known as G language, which has an execution based on dataflow. This means that a function can only be execute when it has available all the data that it needs in its inputs. This permits executing multiple processes in parallel at the same time.

LabVIEW program has two main windows that are completely interrelated: the front panel and the block diagram.

The front panel is the interface that the user manages. It has graphically represented all the inputs and outputs of the program, which can be buttons, text or numerical indicators, lights (LEDs), graphics, images, etc. This allows the user to control the program and to see all the results.

On the other hand, the block diagram is where the program is performed. All the programs implemented in a block diagram have the following elements: data entries represented by controls; functions which perform data operations; and indicators, which are the data outputs. All these elements are connected by wires. The program data circulates through these wires. In the next figure, an example of the two main windows mentioned above can be seen.

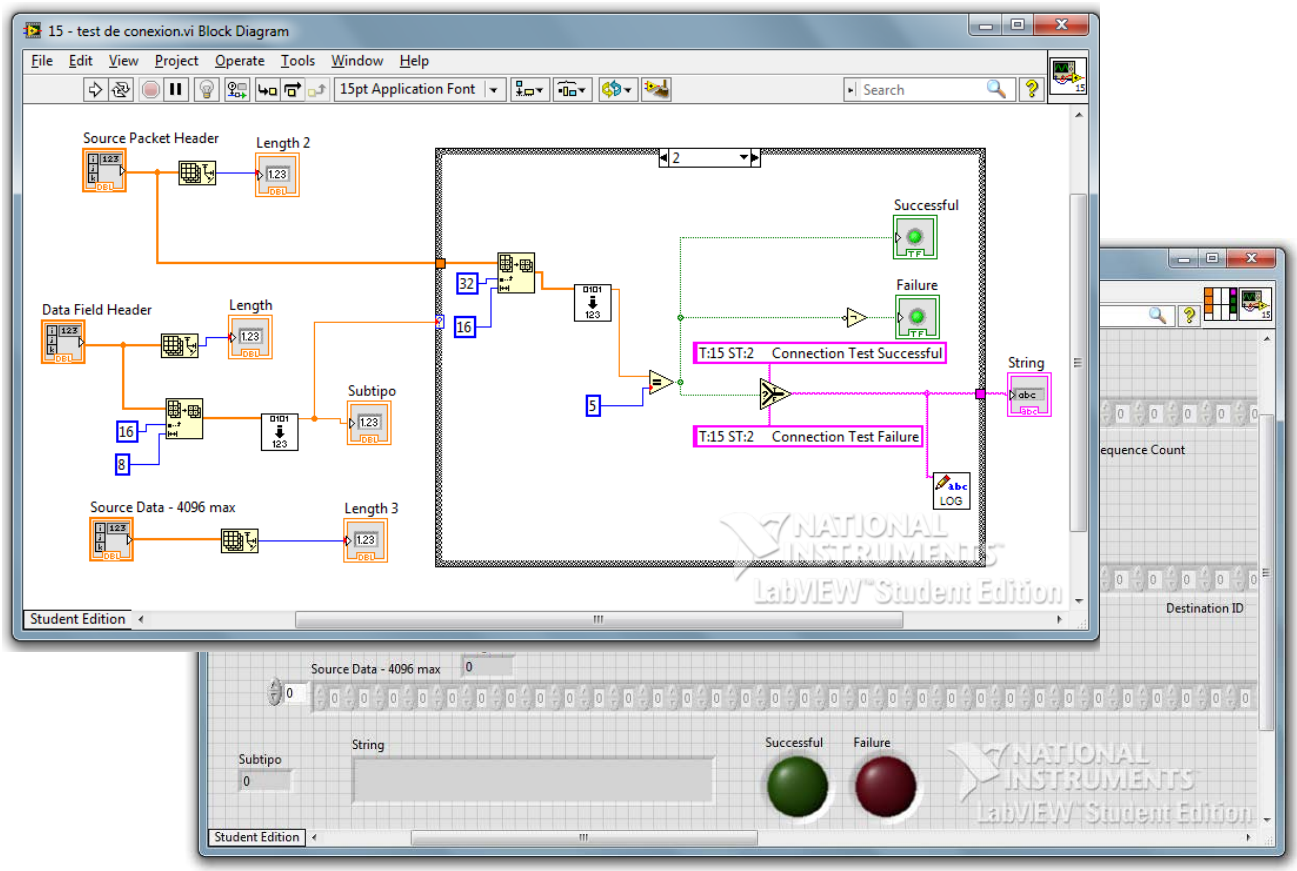


Figure 0.3: Example of the block diagram and the frontal panel

After investigating about all the technology mentioned above –necessary to carry out the project–, the phase of design and development began.

The first one was the design phase. In this phase all the frames and packets formats were planned. These were based on the standard from the ECSS number ECSS-E-70-41A [17]. However, some of the services implemented did not appear in the standard, so they were totally designed by the project team.

After the design, it came the development and implementation phase. In total, there have been designed and implemented 11 services in the communication system. These are very varied, but all of them are necessary. These are the following:

- Telecommand verification service: It notifies if the telecommand sent by the ground station has been accepted/executed or not by the satellite.

- Housekeeping and diagnostic data reporting service: It controls the housekeeping parameters report.

- Event reporting service: It notifies to the user if there has been some error or modification on the satellite.

- Memory management service: It controls everything related to the satellite memory.

- Time management service: It is responsible for managing the time the satellite has on-board.

- On-board operations scheduling service: It manages the on-board telecommands schedule.

- On-board monitoring service: It controls the monitoring of the parameters of the satellite.

- Test service: It permits to check for communication between the satellite and the ground station.

- Temperature sensor instrument: It manages everything related to the temperature of the satellite and it sends the temperature data to the ground station.

- Pictures sensor instrument: It manages everything related to the image sensor (camera) of the satellite and it sends the pictures data to the ground station.

- Reboot service: It permits to reset the satellite system.

After the service implementation, the project passed through a phase of testing and evaluation. In this phase, it was checked that the project accomplished the three main objectives fixed at the beginning of the memory.

Firstly, as for the communication (transmission and reception) between the satellite and the ground station, it was done a lot of tests to calculate the packet loss rate. It was calculated that it had been achieved a packet transmission rate of the 92,2%, so only a 7,8% of the packets were lost.

Secondly, as for the telemetry services implementation, it was also done several tests to check if the program worked correctly. And the results were satisfactory.

And in third place, as for the handiness of the program, it could be seen that the program was very intuitive, simply to use and easy to understand.

So, with all the mentioned above, it could be possible to affirm that the project accomplished all the objectives proposed.

Beside this, at the end of the memory, they were also exposed the future lines of work the project could have. The main lines of further work could be: implement other functionalities using parameters which have been reserved but not used, like the fields of PEC (Packet Error Control), *Version Number*, etc; implement more services both collected or not in the ECSS standard [17]; join the telemetry and telecommand parts in one only program; and increase the number of satellites or ground stations used in the communication.

For all these reasons, it is impossible to say that the project has concluded here. This project has been done always thinking ahead and in the future students. It already has a lot of work, possibilities and improvements to do that the future students will do.

Selected Sections in English

1. Introduction

Nowadays telecommunications are something indispensable. It is impossible to understand a developed society without telecommunications. They allow us to transmit any type of message almost instantaneously. This has led us to a society in which, increasingly, everything tends to be more connected. Concepts like "globalization" or "information society" grow everyday thanks to telecommunications.

A very important part of telecommunications are satellite communications [1]. These allow us to connect faraway places in an almost immediately period of time. From two people speaking by mobile phone, to the communication with the New Horizons probe (which recently sent the first images of Pluto) [18], everything is done thanks to satellite communications.

This Bachelor Thesis focuses on the simulation of one of these satellite communications, especially on a satellite-to-ground communication [1]. To be precise, in this project the Telemetry part of the ground station will be simulated. This terminal is located on the ground and its mission is to communicate with the satellite. The Telemetry part is responsible for receiving and interpreting the data sent from the satellite to the ground. However, all these concepts will be explained in more detail later.

This simulation of a satellite communication will be done using a SDR (Software Defined Radio) platform [4] [5] [6]. This will permit programming the NI USRP (Universal Software Radio Peripheral of National Instruments®) transceivers [7] and they will finally communicate with each other. The peripherals will be programmed using the LabVIEW platform [14] [15] [16]. It is a graphical programming tool which is based on dataflow and which permits simulating wireless communications.

This project will start first with the analysis of the problem statement, considering the state of the art and the socio-economic framework in which it takes place. Then, it will make a little explanation about satellites [1], and a description and analysis of the SDR (Software Defined Radio) systems [4] [5] [6] and of the hardware and software tools used in this project. After that, the designed solution will be proposed and it will do a description thereof. Finally, it will end commenting the results obtained through validation tests, and making a conclusion of the project. The extension of this memory is significant because it has been tried to explain, in detail, the functionality and development of the

implemented system. However, a part of this can be found in the annexes, which provide additional information.

It should be also noted that this project is part of a group of three projects. Each of these projects implements a part of the satellite communication. The present project implements the telemetry functions of the ground station, while the two other projects [2] [3] implement the satellite and the telecommand functions of the ground station respectively. Joining the three projects together a completed satellite communication is finally achieved.

It is noteworthy that, as mentioned at the beginning, the subject matter of this project is totally a current topic. Nowadays it is estimated that there are about 800 active satellites (among government and private) orbiting the Earth [19]. And they make millions of communications daily. In addition, all of them have their respective ground stations which permit sending instructions, controlling and receiving information from the satellite.

For all this, it should be mentioned the importance of this project, because it is about a fully current subject and it will be very useful for the future. Besides, this project will also permit deepening into the subjects studied over the four years of the degree. And it will also permit acquiring new knowledge that has been necessary to carry out this project.

1.1. Motivation

Today, satellite communications have become something basic and transcendental, impossible to remove, and in a continuous development. Besides it is an attractive and fascinating subject that still has a long way to go.

The purpose of this project is, as mentioned above, the simulation of the telemetry part from a ground station in a satellite-to-ground communication [1]. This will allow us to become familiar with this technology and to explore its possibilities.

Besides, the SDR (Software Defined Radio) technology [4] [5] [6] used in this project is not a new technology. However, the recent revolution in digital circuitry has allowed a greatly development and it has become a booming technology. This technology permits recreating radio systems (which are usually implemented with hardware) using software. It makes it a very useful tool when implementing communications systems.

The idea of this project (proposed by the tutor of the project) arose due to the lack of a communications satellite simulator at the University. This one could be used to explain students these subjects in a more attractive way and get a better

assimilation of knowledge doing examples of the operation of the communication system.

The programming and configuration of the entire communications system is a more than laborious work and it would be too much for one person. For this reason, the satellite-to-ground system has been divided in three parts. Each of the parts will be made for each student as his Bachelor Thesis. The three parts the system has been subdivided are:

Firstly, the programming and configuration of the telemetry and telecommand functions of the satellite will be performed by the student Fernando García [3].

And in second and third place, in terms of the ground station, it has been divided according to its two main functions: the telecommand function and the telemetry function. The student Daniel Rodriguez will be responsible for performing the telecommand part [2], while in this current Bachelor Thesis it will perform the telemetry function.

It must be said, however, that although the total work has been divided in three parts, each project will be independent of the other. That is because with this division, the lack of one part will not prejudice the other participants. That is, the three projects will be entirely independent of the others.

1.2. Objectives

The objectives proposed in this project are detailed below. At the end of the project memory (in the testing and validation part) it will be checked whether these objectives have been carried out correctly or not.

As mentioned above, the main objective of this project is developing a simulator of the telemetry function of a ground station EGSE (Electric Ground Support Equipment) that belongs to a satellite communications system [1]. This implementation will be carried out using a SDR platform [4] [5] [6] and a NI USRP [7] transceiver.

Therefore, the final purpose of the work is to achieve a simulation of a complete satellite communication. This will happen when the three aforementioned projects are joined together: the two projects undertaken by the other members of the project team [2] [3], who will carry out the implementation of the entire satellite and the telecommand function of the earth station; and this project.

This project also aims to help future students of the *Telecommunication Systems* subject to have a better understanding of the satellites related issues.

This is because today the University lacks of a simulator of a satellite-ground communications system. When the system has been implemented, it could be set up in class and shown to the students. They will be able to see how the communication between the two terminals occurs. And, in this way, they will be able to understand in a better way how a satellite communications system works and to assimilate knowledge easily.

Due to the latter, the implementation of this simulator should have a simple interface. It has to be easy to understand and manipulate, and it has to allow the users to understand well what is happening at every moment of the communication.

Besides these objectives, a number of other objectives will be also searched, like: delve into the themes studying during the degree, particularly all the satellite related issues studied in the *Telecommunications Systems* subject; research on the technologies used in this Bachelor Thesis, like the SDR (Software Defined Radio) [4] [5] [6] technology, that is now in the process of growth; and increase knowledge on issues related to satellite communications [1], from its history to its way of implementation.

On the other hand, it should be also noted that this project is designed so that future students continue making new implementations and upgrades on it. This is because there are some features in the telemetry systems that won't be possible to carry out due to the lack of time.

1.3. Socio-economic framework

The socio-economic aspects that are affected due to the realization of this project are detailed below. This permits knowing the impact that the project will have, and its feasibility.

Firstly, as mentioned above, the idea of this project was proposed by our tutor and arose due to the lack of a satellite simulator in the Department of Signal Theory and Communications at the Carlos III University of Madrid. Therefore, the project aims to fill a demand for the University. This is because a simulator like this would be very useful for imparting teaching in the center, allowing students to understand better and illustratively how a satellite communication works.

In second place, the fact of developing this project as a Bachelor Thesis allows the University to have a satellite simulator without any extra expense. This is because the University now has all the necessary technology to carry out the project. Besides, the development and implementation work is done by students.

Besides this, the fact of making a satellite simulator for the University by students, permits to adapt the project to the needs demanded. Therefore the simulator will be simple to use and easy to understand allowing future students to

understand how a satellite communication [1] works in a more explanatory and demonstrative way, better than using slides.

On the other hand, this project also opens the possibility to the University to receive commissions or grants for similar projects. This is because so far the University had not done anything like that. If the development of this technology reaches the ears of the companies, they might consider the Carlos III University when they need to do any other similar or related works with this Bachelor Thesis.

Finally, as mentioned above, this project does not end here, because it remains open to future improvements. This will allow future students to continue adding features and improving the simulator to ensure its optimal operation.

1.4. Memory contents

This memory is organized in the easiest and most understandable possible way. It is divided into ten chapters which can be grouped into three sections: a first theoretical block in which the concepts necessary to understand memory and to carry out the implementation are explained; second, a "practical" block where the solution developed for carrying out the implementation is exposed; and finally, a last block in which screening tests are made and conclusions are obtained. The content of the ten chapters of the memory is detailed below:

- Chapter 1: It is the current chapter. It is an introduction to the problem that is going to overcome in this project. This part describes the motivation to do the project, the objectives to be achieved, the socio-economic framework where the impact of the project is analyzed, and the planning thereof.

- Chapter 2: This chapter of the memory is dedicated to the state of the art. It describes the current state of the technology used and it is compared with the technology used to carry out this project.

- Chapter 3: In this chapter a brief theoretical description of the technology used in the Bachelor Thesis is exposed. This is to get a better understanding of the memory by the reader. This chapter will explain both satellite communications [1] and the SDR [4] [5] [6] technology used to carry out the project.

- Chapter 4: This chapter is dedicated to the tools used to carry out the project. It will talk mainly of the NI USRP [7] transceivers used and the LabVIEW [14] [15] [16] program.

- Chapter 5: In this chapter, the proposed solution to achieve the project objective is detailed. It also describes how it has carried out the design

and development. It is the longest chapter of the memory. It explains the modules done with LabVIEW [14] [15] [16] that implement the services of the ground station.

- Chapter 6: This chapter presents the results of the tests. It will finally be compared the implementation done with the objectives set at the beginning of memory.

- Chapter 7: In this chapter, the budget that would be needed to carry out the implementation is detailed.

- Chapter 8: This chapter is perhaps the most important of the memory. In this chapter is where the conclusions after performing all the research and development are discussed. It also explains the future lines of work that could have this project, and the possible improvements that could be made.

- Chapter 9: This chapter is the bibliography that has been necessary to carry out this project.

- Annexes: Several annexes are attached to complete the information contained in the memory.

1.5. Schedule

This section details the planning to carry out the project. Firstly, the activities to be performed and their duration are listed, and then it is illustrated by a Gantt and a PERT diagram.

The project starts on December 4, 2014 and ends on September 23, 2015, representing a total of 9 months and 19 days, i.e. 293 calendar days.

It must be subtracted the non-working days such as weekends, holidays and the summer period when the University is closed (from 6 to 23 of August). Holidays are: 8 and 25 December; 1 and 6 January; March 19; 2 and 3 April; 1 and 2 May; and June 4. Considering all of the above, the total number of working days of the project is reduced to 188 days.

The activities carried out on the project can be classified into four main sections: documentation, implementation, testing and conclusions, and writing the memory. The duration of each of these activities is below:

1. Documentation:
    a. Search for information about satellite communications and study it → 8 days.

b. Search for information about SDR technology and study it → 8 days.

c. Search for information and study about the work environment → 14 days.

d. Analysis of the functionalities to implement → 14 days.

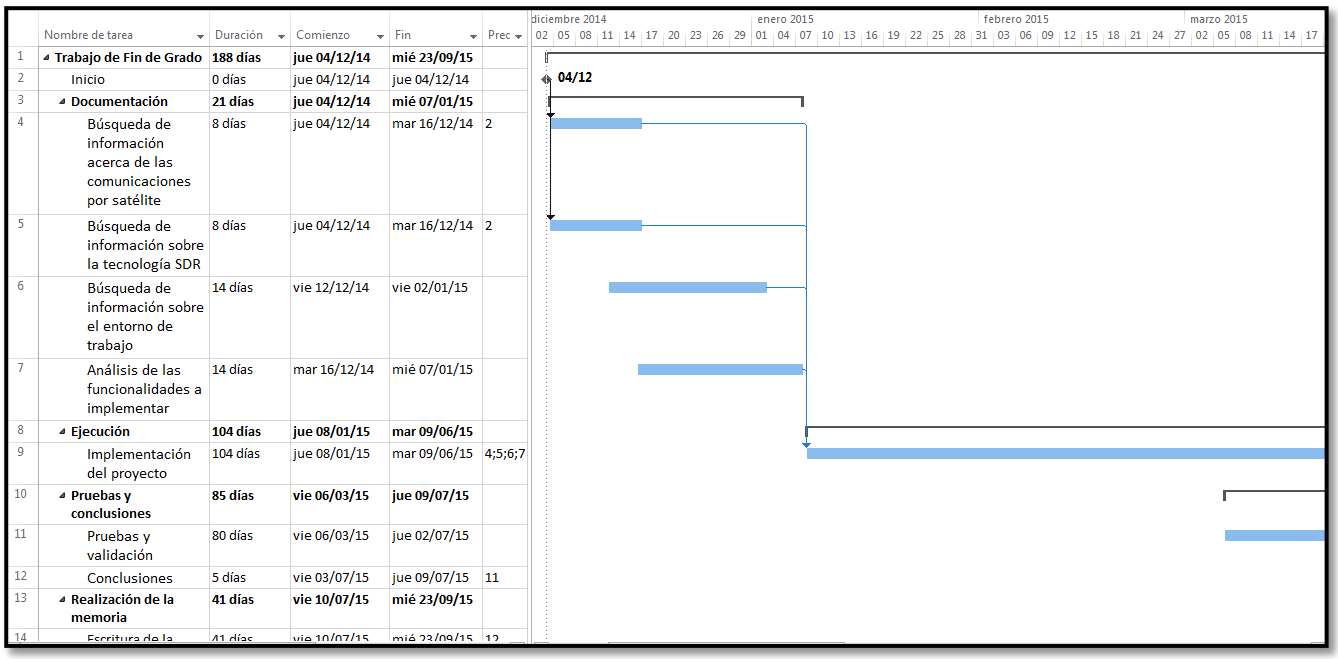
2. Implementation:
   a. Implementation and development of the project → 104 days.

3. Tests and conclusions:
   a. Testing and validation → 80 days.

   b. Drawing conclusions → 5 days.

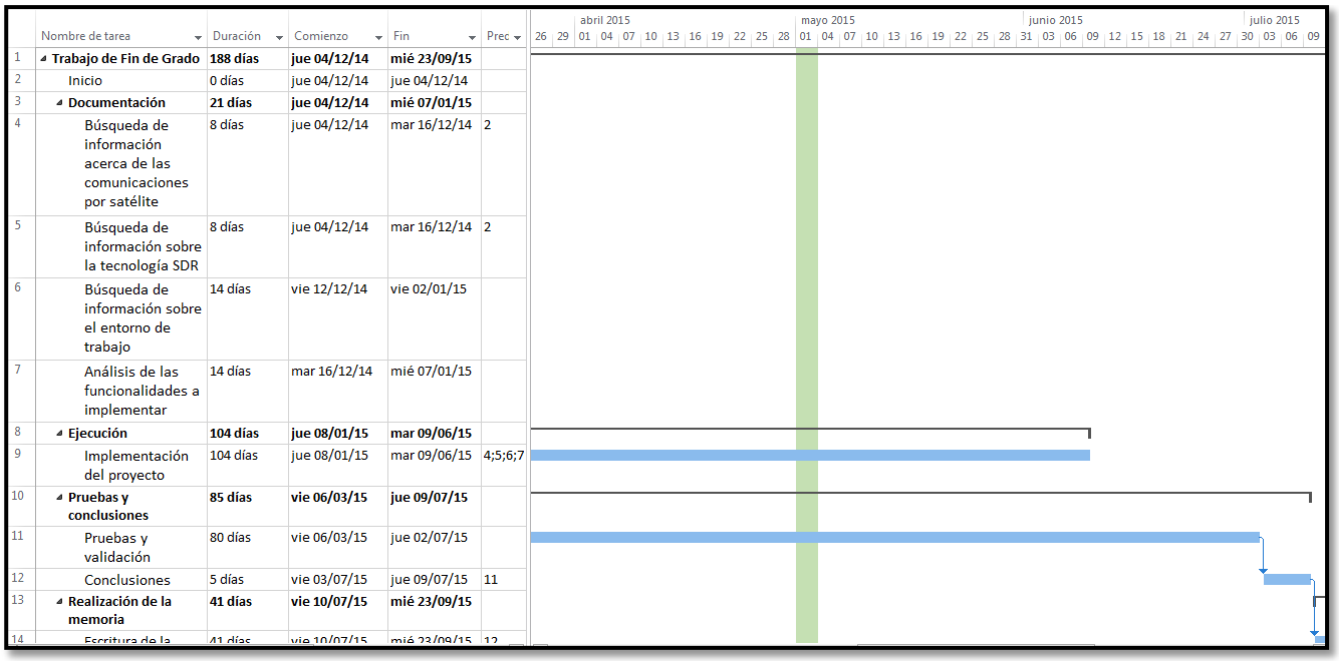
4. Memory:
   a. Writing the memory → 41 days.

This planning of the project is done considering a dedication to the project of about 4 hours per day on average. The Gantt diagram of the project is shown below. In this diagram it is possible to see the duration of each activity, the date of commencement and completion, and the overlaps produced.



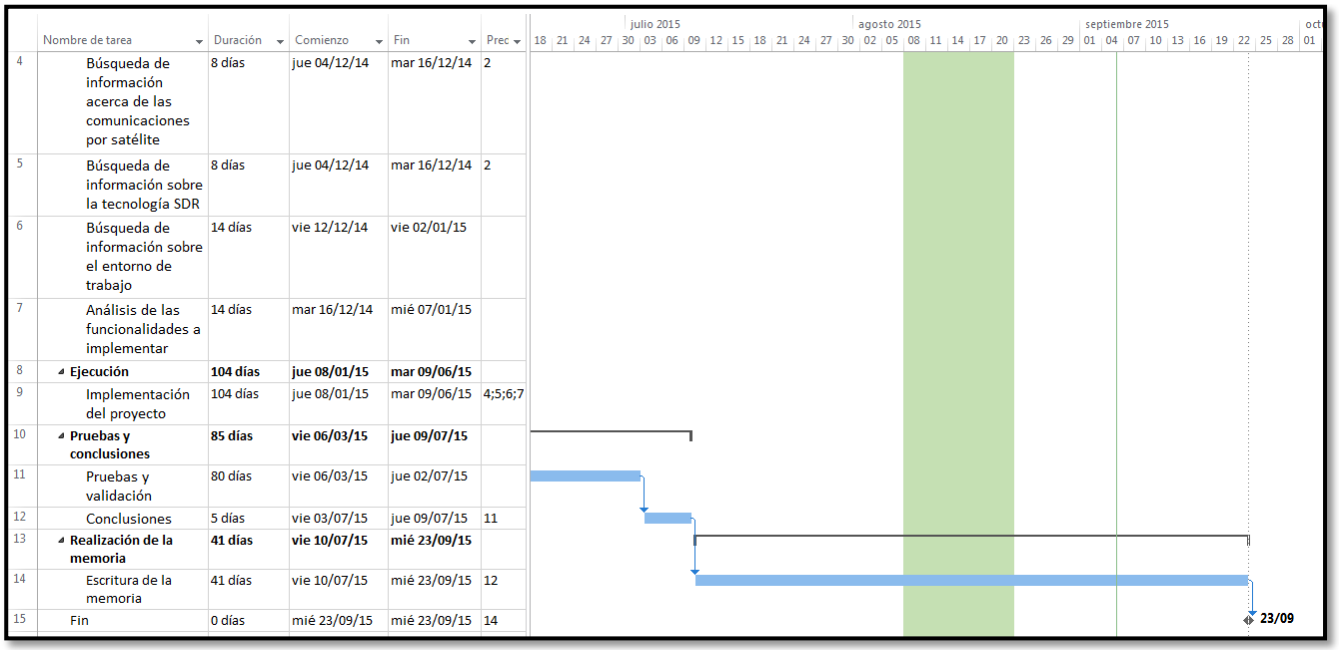
	Nombre de tarea	Duración	Comienzo	Fin	Prec
1	**Trabajo de Fin de Grado**	**188 días**	**jue 04/12/14**	**mié 23/09/15**	
2	Inicio	0 días	jue 04/12/14	jue 04/12/14	
3	**Documentación**	**21 días**	**jue 04/12/14**	**mié 07/01/15**	
4	Búsqueda de información acerca de las comunicaciones por satélite	8 días	jue 04/12/14	mar 16/12/14	2
5	Búsqueda de información sobre la tecnología SDR	8 días	jue 04/12/14	mar 16/12/14	2
6	Búsqueda de información sobre el entorno de trabajo	14 días	vie 12/12/14	vie 02/01/15	
7	Análisis de las funcionalidades a implementar	14 días	mar 16/12/14	mié 07/01/15	
8	**Ejecución**	**104 días**	**jue 08/01/15**	**mar 09/06/15**	
9	Implementación del proyecto	104 días	jue 08/01/15	mar 09/06/15	4;5;6;7
10	**Pruebas y conclusiones**	**85 días**	**vie 06/03/15**	**jue 09/07/15**	
11	Pruebas y validación	80 días	vie 06/03/15	jue 02/07/15	
12	Conclusiones	5 días	vie 03/07/15	jue 09/07/15	11
13	**Realización de la memoria**	**41 días**	**vie 10/07/15**	**mié 23/09/15**	
14	Escritura de la	41 días	vie 10/07/15	mié 23/09/15	12

Figure 0.4: Gantt diagram (part 1)



	Nombre de tarea	Duración	Comienzo	Fin	Prec
1	**Trabajo de Fin de Grado**	**188 días**	**jue 04/12/14**	**mié 23/09/15**	
2	Inicio	0 días	jue 04/12/14	jue 04/12/14	
3	**Documentación**	**21 días**	**jue 04/12/14**	**mié 07/01/15**	
4	Búsqueda de información acerca de las comunicaciones por satélite	8 días	jue 04/12/14	mar 16/12/14	2
5	Búsqueda de información sobre la tecnología SDR	8 días	jue 04/12/14	mar 16/12/14	2
6	Búsqueda de información sobre el entorno de trabajo	14 días	vie 12/12/14	vie 02/01/15	
7	Análisis de las funcionalidades a implementar	14 días	mar 16/12/14	mié 07/01/15	
8	**Ejecución**	**104 días**	**jue 08/01/15**	**mar 09/06/15**	
9	Implementación del proyecto	104 días	jue 08/01/15	mar 09/06/15	4;5;6;7
10	**Pruebas y conclusiones**	**85 días**	**vie 06/03/15**	**jue 09/07/15**	
11	Pruebas y validación	80 días	vie 06/03/15	jue 02/07/15	
12	Conclusiones	5 días	vie 03/07/15	jue 09/07/15	11
13	**Realización de la memoria**	**41 días**	**vie 10/07/15**	**mié 23/09/15**	
14	Escritura de la	41 días	vie 10/07/15	mié 23/09/15	12

Figure 0.5: Gantt diagram (part 2)



	Nombre de tarea	Duración	Comienzo	Fin	Prec
4	Búsqueda de información acerca de las comunicaciones por satélite	8 días	jue 04/12/14	mar 16/12/14	2
5	Búsqueda de información sobre la tecnología SDR	8 días	jue 04/12/14	mar 16/12/14	2
6	Búsqueda de información sobre el entorno de trabajo	14 días	vie 12/12/14	vie 02/01/15	
7	Análisis de las funcionalidades a implementar	14 días	mar 16/12/14	mié 07/01/15	
8	**Ejecución**	**104 días**	**jue 08/01/15**	**mar 09/06/15**	
9	Implementación del proyecto	104 días	jue 08/01/15	mar 09/06/15	4;5;6;7
10	**Pruebas y conclusiones**	**85 días**	**vie 06/03/15**	**jue 09/07/15**	
11	Pruebas y validación	80 días	vie 06/03/15	jue 02/07/15	
12	Conclusiones	5 días	vie 03/07/15	jue 09/07/15	11
13	**Realización de la memoria**	**41 días**	**vie 10/07/15**	**mié 23/09/15**	
14	Escritura de la memoria	41 días	vie 10/07/15	mié 23/09/15	12
15	Fin	0 días	mié 23/09/15	mié 23/09/15	14

Figure 0.6: Gantt diagram (part 3)

Besides this, Figure 1.4 also shows the PERT diagram of the project. It shows the order of precedence of the activities and the critical route of the project, outlined in blue.

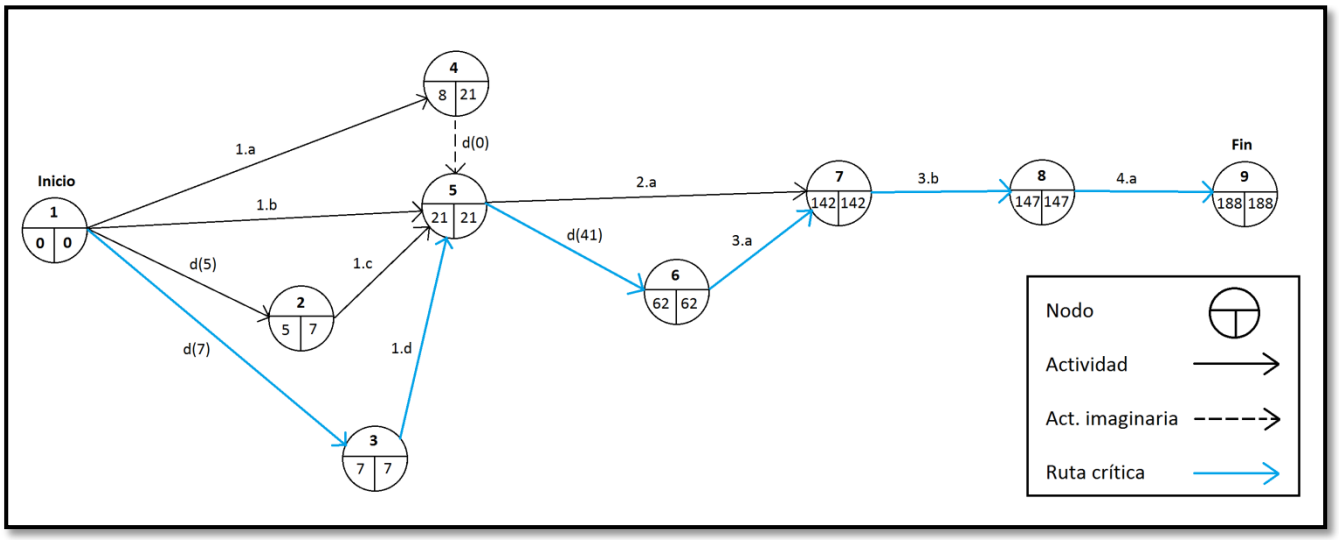


Figure 0.7: PERT diagram

As seen in the picture above, the total project duration is 188 days. It is equivalent to 27 weeks approximately.

2. Conclusions and further work

This chapter presents, firstly, the general conclusions of the project and, secondly, the future lines of work for which the project could be continued.

2.1. Conclusions

As mentioned above, this project arose due to the lack of a simulator of a communication satellite in the Carlos III University of Madrid. The idea was proposed by the tutor of the project and it has been carried out by three students of the University.

When the project began, in the planning phase, a series of objectives that the project should achieve were raised. These were: firstly, and as the common objective of the three students of the project, to be able to execute a simulation of a satellite communication implemented in a structure defined by software (SDR); secondly, and as an objective only of this project, to implement successfully the telemetry functionalities of a ground station which belong to a satellite

communication; and thirdly, to make a simple program, easy to operate and easy to understand.

Besides these objectives, it was also sought to deepen in the subjects studied in the degree (particularly all the satellite related issues), research on technologies that it had been used (SDR, LabVIEW...) and thus broaden the knowledge on this field of the telecommunications.

Related to the latter, before the implementation of the project began, it was necessary to do a research work, which is described in Chapters 2, 3 and 4 of the memory.

First, an investigation about the technologies used in this project was done. These were the satellite communications [1] and the SDR technology [4] [5] [6].

As for the first, a lot of information was sought about what satellite communications are [1], about its history, and about the two main elements in a satellite communication: the satellite and the ground station. The concepts of telemetry and telecommand were also defined because they are widely used in this project.

And as for the SDR (Software Defined Radio) technology [4] [5] [6], it was also done a research process about what it is, its evolution, and how and how much is currently used this technology.

In second place, it was also necessary to investigate about the working environment of the project. It was essential to search information about the two main tools used to carry out the project: the NI USRP 2920 transceiver and the LabVIEW programming environment.

About the NI USRP [7] – [10] transceiver, information was sought about the operating characteristics thereof. And as for the LabVIEW [14] [15] [16] programming environment, manuals were consulted about how to use it and how to make programs with it.

And thirdly, an investigation about similar projects that already exist was done. They were searched both in the market and in universities, and a comparison among them and this project was done. This way, it served to see how this project and the existing ones differ and what the highlights of this project are.

After this research process, the next step was to design and develop the program. The program was conducted entirely in the LabVIEW platform [14] [15] [16] and it was implemented on a NI USRP 2920 transceiver [7] – [10]. However, throughout the development of the program, several complications and several design alternatives arose, as mentioned in chapter 6. Nevertheless, the project was successfully completed.

After that, the project went through a period of testing and evaluation. The purpose was to see whether it met the objectives set at the beginning of the project. After doing that, it was seen that the project successfully implemented the property features of the ground station, that it properly communicated with the satellite, and that the program was simple and easy to use. Therefore it was found that the project had successfully accomplished the objectives set at the beginning.

Consequently, it can be said that both educational and technical objectives of the project have been achieved successfully, because it has been created a very visual simulator that will help to explain the students these subjects in a much easier and more attractive way. Besides, the communication between the ground station and the satellite conforming with current regulations has been achieved.

Finally, just say that the project has contributed much to the student who has made it. Firstly, it has permitted to do a great expansion of knowledge possessed by the student about satellites related issues. Secondly, it has also allowed the student to learn to use tools that were completely unknown for him, as the LabVIEW platform or NI USRP transceivers. And thirdly, the fact of doing this project together has increased the student's ability to work in teams.

2.2. Further work

As mentioned throughout the memory, this project has been designed constantly thinking ahead. It is a project that is designed to help teaching at the University. For this reason it is also designed so that the following generations continue developing and improving it.

The future lines of work that this project could have are the following:

Firstly, it could be implemented some parameters or functionalities that it have not been able to implement in this project, mainly because of the lack of time. However, they have been taken into consideration during the design and development of the project.

An example of this is the PEC (Packet Error Control) field of the telemetry packet (see section 5.1). This field is used to detect faults in the packet. It allows the ground station to verify the integrity of the entire message. The implementation of this functionality could be done using checksum verification, a CRC code, etc.

However, despite the fact that this has not been implemented, it has been considered during the carrying out of the program. These 16 bits from the PEC are read in the program, but no action is taken with them.

Besides this, other packet could be also implemented with other format but without losing the current. This is possible because, as mentioned in section 5.1, there is a field in the packet header called *Version Number*. This field indicates the version number of the packet. This first version of the packages that has been made in the project has assigned the number 0. So, if they wanted to add other versions, they only would have to change the version number in the packet and implement a system with the new packet format. Moreover, thanks to the *Version Number* field, a transmission of packets from multiple versions could be even achieved.

Secondly, another future line of work could be to implement more services and functionalities of the ground station. In the standard from the ECSS number ECSS-E-70-41A [17] (in which this project is based) are collected much more services that it could not be implemented for the lack of time.

Furthermore, in addition to the services covered by the standard [17], it is also possible to implement other specific services that are not included in the standard, as services 32 and 64 of this project. This opens an almost infinite range of new functionalities and services that could be implemented on the ground station.

In third place, as mentioned in section 6.1, another possible improvement could be to join the functions of telemetry and telecommand from the earth station in a single program with communication between both parts. This would open the possibility of making a system of sending and receiving packets with acknowledgment (ACK). This means that whenever the ground station would receive a packet from the satellite, it would send back an acknowledgment packet. If after a time the satellite did not receive this package, it would send the information again, and the same from the ground station.

And fourth and finally, another future line of work could be increasing the number of satellites or ground stations simulated. This improvement has also been considered since the beginning of the project. For this reason, a field has been reserved in the header of the data (Data Field Header), named *Destination ID*. This field is used to indicate who the packet is for, in case there is more than one ground station or more than one satellite. This improvement would be easy to implement but, like the previous cases, it could not be performed due to time constraints.

As it can be seen, this project does not end here, because it still has a wide range of possibilities opened and infinity of new versions and improvements of the project. There is therefore a long way ahead to go, which future students form future generations will travel.

1. Introducción

Hoy en día las telecomunicaciones son algo indispensable, algo sin lo que sería imposible entender una sociedad desarrollada. Permiten transmitir cualquier mensaje de cualquier tipo de forma casi instantánea, lo que nos ha llevado a una sociedad en la que, cada vez más, todo tiende a estar más conectado. Conceptos como la "globalización" o la "sociedad de la información" crecen cada día gracias a las telecomunicaciones.

Una parte más que importante de las telecomunicaciones son las comunicaciones por satélite [1]. Éstas permiten conectar lugares a gran distancia en un período de tiempo casi inmediato. Desde dos personas hablando mediante telefonía móvil, hasta la comunicación con la sonda New Horizons (que hace poco envió las primeras imágenes de Plutón) [18], se realiza todo mediante comunicación por satélites.

Este trabajo de fin de grado (TFG) se centra en la simulación de una de estas comunicaciones por satélite, especialmente, en una comunicación satélite-tierra [1]. Concretamente, en este proyecto se simulará la parte de Telemetría de la estación de tierra, que es el terminal situado en la tierra encargado de comunicarse con el satélite. La parte de Telemetría consistirá en recibir e interpretar los datos enviados desde el satélite a la tierra. De todas formas, todos estos conceptos se explicarán más adelante de forma más detallada.

Esta simulación de una comunicación por satélite se hará mediante el uso de una plataforma SDR (Software Defined Radio) [4] [5] [6], que permitirá programar los transceptores NI USRP (Universal Software Radio Peripheral de National Instruments®) [7] que serán los que finalmente se comuniquen entre sí. La programación de los periféricos se realizará mediante la plataforma LabVIEW [14] [15] [16], una herramienta de programación gráfica basada en flujo de datos que permite simular comunicaciones inalámbricas.

El presente proyecto comenzará en primer lugar con el análisis del planteamiento del problema, teniendo en cuenta el estado del arte y el marco socio-económico en el que se lleva a cabo. Posteriormente se hará una pequeña mención explicativa sobre los satélites [1], y una descripción y análisis de forma más extendida de los sistemas SDR (Software Defined Radio) [4] [5] [6] y de las herramientas hardware y software utilizadas. Después se propondrá la solución diseñada y se hará una descripción detallada de ésta. Finalmente se terminará comentando los resultados obtenidos mediante pruebas de validación, y haciendo una conclusión del proyecto. La extensión de esta memoria es considerable debido a que se ha querido exponer, de forma detallada, las funcionalidades y el desarrollo del sistema implementado. No obstante, una parte de esto se encuentra en los anexos, que aportan información adicional.

Se ha de aclarar también que este proyecto forma parte de un conjunto de tres proyectos encargados cada uno de implementar una parte de la comunicación por satélite. En este proyecto se han implementado las funciones de telemetría de la estación de tierra, mientras que los otros dos proyectos [2] [3] implementan el satélite y las funciones de telecomandos de la estación de tierra respectivamente. Al juntar los tres proyectos se consigue simular una comunicación por satélite completa.

Cabe destacar que, como se ha comentado al principio, el tema de estudio del presente proyecto es un tema totalmente actual. Hoy en día se calcula que hay alrededor de 800 satélites activos (entre gubernamentales y privados) orbitando alrededor de la Tierra [19] y por los que se realizan millones de comunicaciones al día. Además, todos ellos cuentan con sus respectivas estaciones de tierra que se encargan de enviar instrucciones, controlar y recibir información del satélite.

Por todo esto, cabe mencionar la importancia de este proyecto, tanto por ser un tema de trabajo totalmente actual, como por su utilidad para el futuro. Asimismo, el presente proyecto permitirá también profundizar en los temas estudiados a lo largo de estos cuatro años de grado, además de adquirir nuevos conocimientos que han sido necesarios para llevar a cabo este trabajo.

1.1. Motivación

A día de hoy las comunicaciones por satélite se han convertido en algo básico y trascendental, imposible de suprimir, y en continuo desarrollo. Además es un tema atrayente y fascinante que tiene todavía un largo camino por delante.

La finalidad de este proyecto será, como se ha comentado anteriormente, realizar la simulación de la parte de telemetría de una estación de tierra en una comunicación satélite-base [1], lo que nos permitirá familiarizarnos con esta tecnología, además de explorar sus posibilidades.

Además de esto, la tecnología SDR (Software Defined Radio) [4] [5] [6] que se va a emplear es de igual modo una tecnología que, aunque no es nueva, la reciente revolución en cuanto a la circuitería digital ha permitido que se desarrolle enormemente y se ha convertido en una tecnología en auge. Esta tecnología permite recrear mediante software sistemas de radiocomunicaciones que suelen ser implementados en tecnología hardware, lo que la convierte en una herramienta de gran utilidad a la hora de implementar sistemas de comunicaciones.

La idea de este proyecto, propuesta por el tutor del proyecto, surge debido a la falta en la Universidad de un simulador de comunicaciones por satélite, que se podría utilizar para explicar a los alumnos estos temas de una forma más atractiva

y conseguir, mediante ejemplos del funcionamiento del sistema de comunicaciones, una mejor asimilación de los conocimientos.

La programación y configuración de este sistema completo de comunicaciones es un trabajo más que laborioso y sería excesivo para una sola persona. Por este motivo, se ha dividido la configuración del sistema satélite-base en tres partes, las cuales serán realizadas cada una por un alumno como su trabajo de fin de grado. Las tres partes en las que se ha subdividido el sistema son las siguientes:

En primer lugar, el satélite, cuya programación y configuración de las funciones de telemetría y telecomandos serán realizadas por el alumno Fernando García [3].

Y en segundo y tercer lugar, en cuanto a la estación de tierra, se ha dividido su programación de acuerdo a sus dos funcionalidades principales: la función de telecomandos y la función de telemetría. El alumno Daniel Rodríguez será el encargado de realizar la parte de telecomandos [2], mientras que en el presente TFG se realizará la parte de telemetría.

Se ha de decir, sin embargo, que aunque se ha dividido el trabajo total en tres partes, los proyectos de cada alumno serán independientes de los demás, ya que con la división que se ha realizado, la falta de una de las partes no perjudicará a los otros participantes. Es decir, los tres proyectos serán trabajos totalmente independientes.

1.2. Objetivos

A continuación se detallarán los objetivos propuestos a la hora de llevar a cabo este TFG. Finalmente, en el apartado de Pruebas y Validación, se comprobará si estos objetivos han sido llevados a cabo correctamente o no.

Como se ha comentado anteriormente, el principal objetivo de este proyecto es desarrollar la simulación de la parte correspondiente a la función de telemetría de una estación de tierra EGSE (Electric Ground Support Equipment) perteneciente a un sistema de comunicaciones por satélite [1]. Esta implementación se llevará a cabo mediante una plataforma SDR [4] [5] [6] y utilizando un transceptor NI USRP [7].

El propósito final del trabajo será por tanto conseguir una simulación de una comunicación por satélite completa al juntar los tres proyectos mencionados anteriormente: los dos proyectos realizados por los otros integrantes del equipo del proyecto [2] [3], encargados de llevar a cabo la implementación del satélite y de la función de telecomandos de la estación de tierra; y el presente proyecto.

La idea a la hora de llevar a cabo esta implementación también tiene como objetivo el poder ayudar a los futuros alumnos de la asignatura de *Sistemas de Telecomunicación* a comprender mejor los temas relacionados con satélites, ya que hoy en día la Universidad carece de un simulador de un sistema de comunicaciones satélite-tierra. Una vez implementado el sistema, se podría instalar en clase y mostrar a los alumnos cómo se produce la comunicación entre los dos terminales para que, de este modo, éstos puedan entender mejor cómo funciona un sistema de comunicaciones por satélite y asimilen mejor los conocimientos.

Debido a esto último, la implementación de este simulador deberá tener una interfaz sencilla, que sea fácil de entender y manipular, y que permita a los usuarios comprender bien que es lo que está sucediendo en cada momento de la comunicación.

Además de estos objetivos, también se buscarán otra serie de objetivos, como: profundizar en los temas trabajados durante el grado, sobre todo lo relativo a los satélites, estudiados en la asignatura de *Sistemas de Telecomunicación*; investigar sobre las tecnologías utilizadas en este TFG, como la tecnología SDR (Software Defined Radio) [4] [5] [6] que ahora mismo es una tecnología en pleno proceso de crecimiento; y ampliar conocimientos en los temas relacionados con las comunicaciones por satélite [1], desde su historia hasta su forma de implementación.

Por otro lado, también cabe mencionar que este proyecto está pensado para que en el futuro se continúen haciendo nuevas implementaciones y mejoras sobre él, ya que hay algunas funcionalidades de los sistemas de telemetría que no se podrán llevar a cabo por falta de tiempo.

1.3. Marco socio-económico

A continuación se detallan los aspectos socio-económicos que se ven afectados debido a la realización de este proyecto. Esto permite conocer tanto el impacto que va a tener la realización del proyecto, como la factibilidad del mismo.

En primer lugar, y como se comentó anteriormente, la idea de este proyecto fue propuesta por nuestro tutor y surgió debido a que el departamento de Teoría de la Señal y Comunicaciones de la Universidad Carlos III de Madrid no posee actualmente un simulador de satélite. Por este motivo, el proyecto pretende cubrir una demanda de la Universidad, ya que un simulador de estas características sería muy útil a la hora de impartir la docencia en el centro, permitiendo a los alumnos comprender mejor y de forma ilustrativa cómo funciona una comunicación por satélite.

En segundo lugar, el hecho de desarrollar este proyecto como Trabajo de Fin de Grado (TFG) permite dotar a la Universidad de un simulador de satélite sin que ésta tenga que realizar ningún gasto extraordinario, ya que la Universidad cuenta actualmente con toda la tecnología necesaria para llevar a cabo el proyecto, y además el trabajo de implementación y desarrollo es llevado a cabo por alumnos.

Además de esto, el hecho de realizar un simulador de satélite para la Universidad por parte de alumnos, permite adaptar el proyecto a las necesidades demandadas. Por este motivo el simulador será sencillo de usar y fácil de entender permitiendo a los futuros estudiantes comprender cómo funciona una comunicación por satélite [1] de una forma mucho más aclaratoria y demostrativa que mediante el uso de diapositivas.

Por otro lado, la realización de este proyecto abre también la posibilidad a la Universidad de recibir encargos o subvenciones para otros proyectos similares, ya que hasta ahora no se había realizado nada semejante en la Universidad. Si el desarrollo de este tipo de tecnología llega a oídos de las empresas, éstas podrían tener en cuenta a la Universidad Carlos III a la hora de llevar a cabo algún otro proyecto similar o relacionado con el presente trabajo de fin de grado.

Finalmente, como se mencionó anteriormente, este proyecto no finaliza aquí, sino que queda abierto a futuras mejoras, lo que permitirá a los estudiantes venideros seguir añadiendo funcionalidades y perfeccionar el simulador para conseguir que su funcionamiento sea óptimo.

1.4. Contenido de la memoria

La presente memoria está organizada de tal forma que su lectura sea lo más fácil y comprensible posible. Se divide en diez capítulos que pueden ser agrupados en tres grandes bloques: un primer bloque teórico en el que se explican los conceptos necesarios para entender la memoria y para llevar a cabo la implementación; en segundo lugar, un bloque "práctico" en el que se expone la solución desarrollada para llevar a cabo la implementación; y finalmente, un último bloque en el que se realizan pruebas de evaluación y se obtienen conclusiones. El contenido de los diez capítulos de la memoria se detalla a continuación:

- Capítulo 1: Es el capítulo en el que se encuentra actualmente. En él se hace una introducción al problema que se quiere solventar en este proyecto. En esta parte se describen la motivación por la que se ha llevado a cabo el proyecto, el objetivo que se pretende alcanzar, el marco socio-económico que se analiza el impacto que tiene la realización del proyecto, y la planificación del mismo.

- Capítulo 2: Este capítulo de la memoria está dedicado al estado del arte. En él se retrata la situación actual de la tecnología empleada y se compara con la que se ha utilizado para llevar a cabo el presente proyecto.

- Capítulo 3: En este capítulo se hace una breve descripción teórica de la tecnología utilizada en el TFG para conseguir una mejor comprensión del mismo por parte del lector. En este capítulo se hablará tanto de las comunicaciones por satélite [1] como de la tecnología SDR [4] [5] [6] empleada para desarrollar la implementación.

- Capítulo 4: Este capítulo está dedicado a las herramientas usadas para llevar a cabo el proyecto. Se hablará principalmente de los NI USRP [7] utilizados y del programa LabVIEW [14] [15] [16].

- Capítulo 5: En este capítulo se detalla la solución propuesta para conseguir el objetivo marcado. También se describe cómo se ha llevado a cabo el diseño y el desarrollo del mismo. Es el capítulo más extenso de la memoria ya que en él se explican los módulos realizados con LabVIEW [14] [15] [16] que implementan los distintos servicios que ha de realizar la estación de tierra.

- Capítulo 6: Este capítulo está orientado a la presentación de los resultados de las pruebas realizadas. En él se verá finalmente si la implementación realizada se ajusta y cumple los objetivos propuestos al principio de la memoria.

- Capítulo 7: En este capítulo se detalla el presupuesto que sería necesario para llevar a cabo la implementación diseñada en el TFG.

- Capítulo 8: Este capítulo es quizá el más importante de la memoria ya que es donde se exponen las conclusiones obtenidas después de realizar toda la investigación y desarrollo. También se explican en este capítulo las futuras líneas de trabajo que podría tener este TFG y qué posibles mejoras se podrían realizar.

- Capítulo 9: Este capítulo contiene la bibliografía que ha sido necesaria para llevar a cabo este proyecto.

- Anexos: Se adjuntan varios anexos para completar la información expuesta en la memoria.

1.5. Cronograma

En este apartado se detalla la planificación prevista para llevar a cabo el proyecto. Primero se detallan las actividades a realizar y su duración, y después se ilustra mediante un diagrama de Gantt y un diagrama PERT.

El proyecto comienza el 4 de diciembre de 2014 y finaliza el 23 de septiembre de 2015, lo que supone un total de 9 meses y 19 días, es decir, 293 días naturales. A este número de días hay que restarle los días no laborables como fines de semana y días festivos, así como el período de verano en el que la Universidad permanece cerrada (del 6 al 23 de agosto). Los días festivos son: 8 y 25 de diciembre; 1 y 6 de enero; 19 de marzo; 2 y 3 de abril; 1 y 2 de mayo; y 4 de junio. Contando con todo lo anterior, el número total de días laborables del proyecto se reduce a 188 días.

Las actividades llevadas a cabo en el proyecto pueden ser clasificadas en cuatro grandes bloques: documentación, ejecución, pruebas y conclusiones, y escritura de la memoria. La duración de cada una de estas actividades se detalla a continuación:

1. Documentación:
    a. Búsqueda de información acerca de las comunicaciones por satélite y estudio de la misma → 8 días.

    b. Búsqueda de información sobre la tecnología SDR y estudio de la misma → 8 días.

    c. Búsqueda de información y estudio sobre el funcionamiento del entorno de trabajo → 14 días.

    d. Análisis de las funcionalidades a implementar → 14 días.

2. Ejecución:
    a. Implementación y desarrollo del proyecto → 104 días.

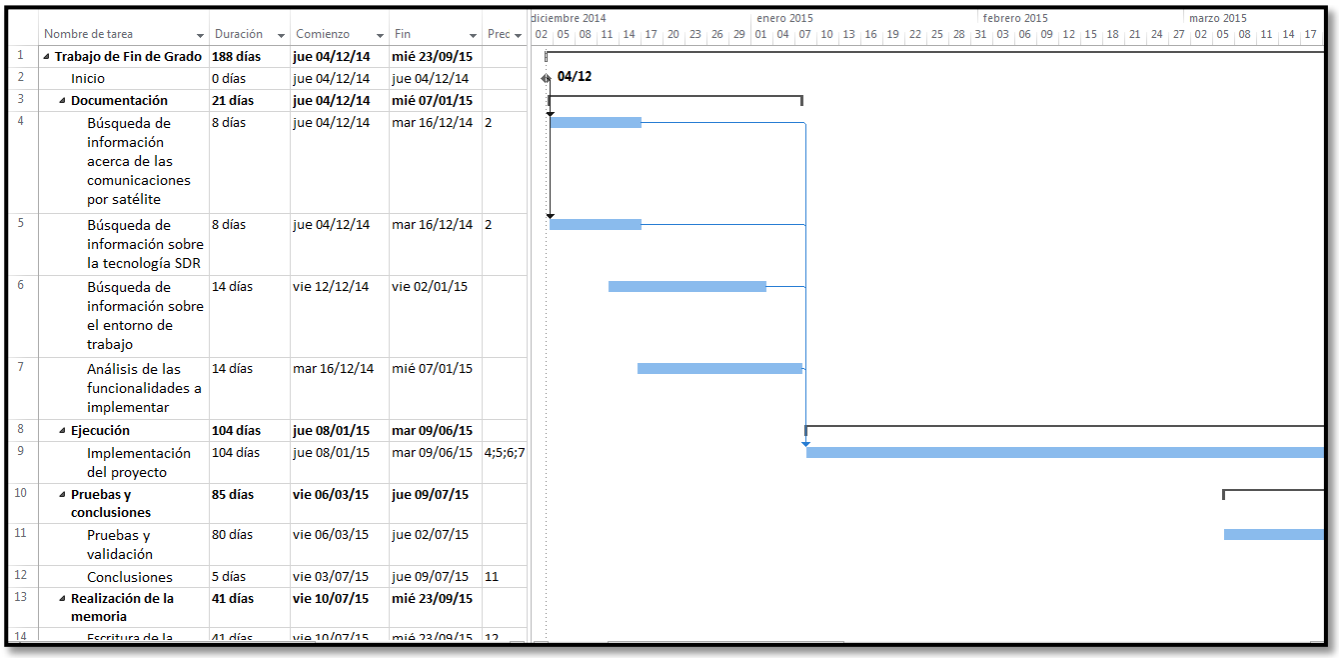
3. Pruebas y conclusiones:
    a. Realización de pruebas y validación → 80 días.

    b. Obtención de conclusiones → 5 días.

4. Realización de la memoria:
    a. Escritura de la memoria → 41 días.

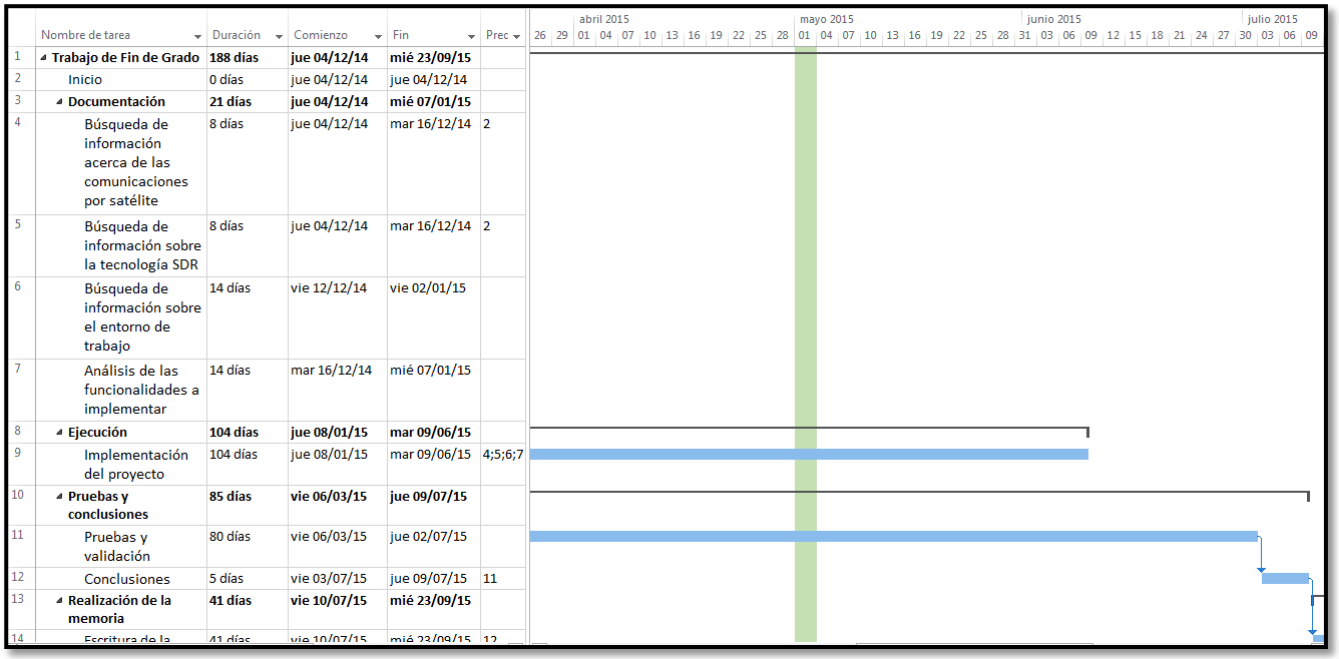
Esta planificación del proyecto se ha llevado a cabo contando con una dedicación al proyecto de unas 4 horas al día de media. A continuación se muestra el diagrama de Gantt del proyecto, donde se puede ver, además de la duración de

cada actividad, la fecha de comienzo y finalización de cada una, y los solapamientos producidos.



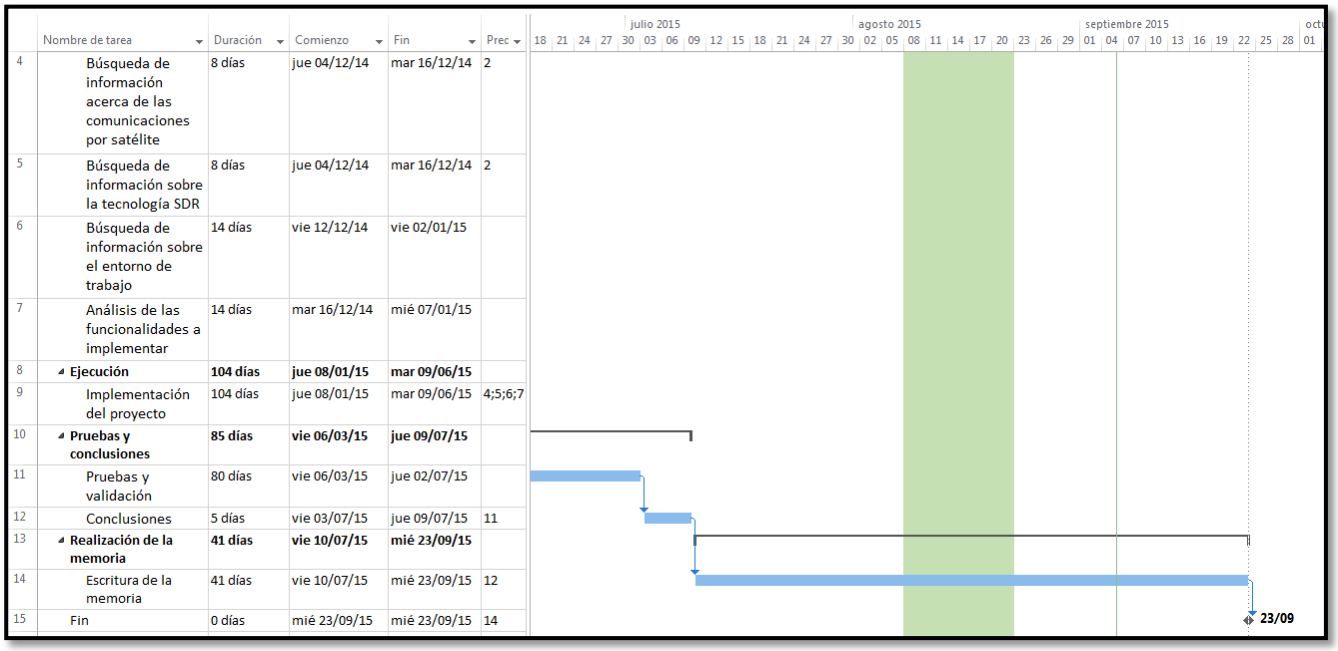
	Nombre de tarea	Duración	Comienzo	Fin	Prec
1	**Trabajo de Fin de Grado**	**188 días**	**jue 04/12/14**	**mié 23/09/15**	
2	Inicio	0 días	jue 04/12/14	jue 04/12/14	
3	**Documentación**	**21 días**	**jue 04/12/14**	**mié 07/01/15**	
4	Búsqueda de información acerca de las comunicaciones por satélite	8 días	jue 04/12/14	mar 16/12/14	2
5	Búsqueda de información sobre la tecnología SDR	8 días	jue 04/12/14	mar 16/12/14	2
6	Búsqueda de información sobre el entorno de trabajo	14 días	vie 12/12/14	vie 02/01/15	
7	Análisis de las funcionalidades a implementar	14 días	mar 16/12/14	mié 07/01/15	
8	**Ejecución**	**104 días**	**jue 08/01/15**	**mar 09/06/15**	
9	Implementación del proyecto	104 días	jue 08/01/15	mar 09/06/15	4;5;6;7
10	**Pruebas y conclusiones**	**85 días**	**vie 06/03/15**	**jue 09/07/15**	
11	Pruebas y validación	80 días	vie 06/03/15	jue 02/07/15	
12	Conclusiones	5 días	vie 03/07/15	jue 09/07/15	11
13	**Realización de la memoria**	**41 días**	**vie 10/07/15**	**mié 23/09/15**	
14	Escritura de la	41 días	vie 10/07/15	mié 23/09/15	12

Figura 1.1: Diagrama de Gantt (parte 1)



	Nombre de tarea	Duración	Comienzo	Fin	Prec
1	**Trabajo de Fin de Grado**	**188 días**	**jue 04/12/14**	**mié 23/09/15**	
2	Inicio	0 días	jue 04/12/14	jue 04/12/14	
3	**Documentación**	**21 días**	**jue 04/12/14**	**mié 07/01/15**	
4	Búsqueda de información acerca de las comunicaciones por satélite	8 días	jue 04/12/14	mar 16/12/14	2
5	Búsqueda de información sobre la tecnología SDR	8 días	jue 04/12/14	mar 16/12/14	2
6	Búsqueda de información sobre el entorno de trabajo	14 días	vie 12/12/14	vie 02/01/15	
7	Análisis de las funcionalidades a implementar	14 días	mar 16/12/14	mié 07/01/15	
8	**Ejecución**	**104 días**	**jue 08/01/15**	**mar 09/06/15**	
9	Implementación del proyecto	104 días	jue 08/01/15	mar 09/06/15	4;5;6;7
10	**Pruebas y conclusiones**	**85 días**	**vie 06/03/15**	**jue 09/07/15**	
11	Pruebas y validación	80 días	vie 06/03/15	jue 02/07/15	
12	Conclusiones	5 días	vie 03/07/15	jue 09/07/15	11
13	**Realización de la memoria**	**41 días**	**vie 10/07/15**	**mié 23/09/15**	
14	Escritura de la	41 días	vie 10/07/15	mié 23/09/15	12

Figura 1.2: Diagrama de Gantt (parte 2)



	Nombre de tarea	Duración	Comienzo	Fin	Prec
4	Búsqueda de información acerca de las comunicaciones por satélite	8 días	jue 04/12/14	mar 16/12/14	2
5	Búsqueda de información sobre la tecnología SDR	8 días	jue 04/12/14	mar 16/12/14	2
6	Búsqueda de información sobre el entorno de trabajo	14 días	vie 12/12/14	vie 02/01/15	
7	Análisis de las funcionalidades a implementar	14 días	mar 16/12/14	mié 07/01/15	
8	**Ejecución**	**104 días**	**jue 08/01/15**	**mar 09/06/15**	
9	Implementación del proyecto	104 días	jue 08/01/15	mar 09/06/15	4;5;6;7
10	**Pruebas y conclusiones**	**85 días**	**vie 06/03/15**	**jue 09/07/15**	
11	Pruebas y validación	80 días	vie 06/03/15	jue 02/07/15	
12	Conclusiones	5 días	vie 03/07/15	jue 09/07/15	11
13	**Realización de la memoria**	**41 días**	**vie 10/07/15**	**mié 23/09/15**	
14	Escritura de la memoria	41 días	vie 10/07/15	mié 23/09/15	12
15	Fin	0 días	mié 23/09/15	mié 23/09/15	14

Figura 1.3: Diagrama de Gantt (parte 3)

Además de esto, en la Figura 1.4 se muestra también el diagrama PERT del proyecto. En él se puede ver claramente el orden de precedencia de las actividades y la ruta crítica del proyecto, señalada en color azul.

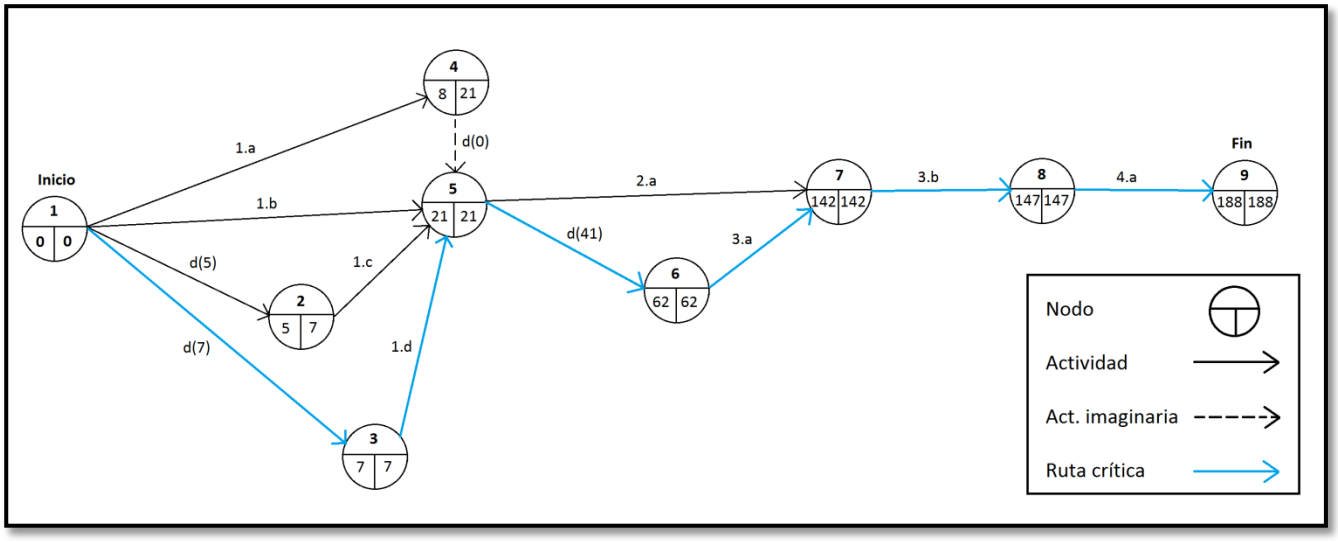


Figura 1.4: Diagrama PERT

Como se puede observar en la imagen anterior, la duración total del proyecto es de 188 días, lo que equivale a unas 27 semanas aproximadamente.

2. Estado del arte

En este capítulo se expone el estado del arte en el que se ha llevado a cabo este proyecto. La finalidad de este capítulo es investigar sobre la tecnología similar a la realizada en el proyecto existente en el momento en el que se ha llevado a cabo, y compararla.

Como se comentó anteriormente, en este proyecto se desarrolla un simulador de una comunicación por satélite. Esta tecnología es totalmente actual ya que hoy en día está en pleno desarrollo, lo que provoca que esté en constante cambio y evolución.

En primer lugar, como se dijo en el capítulo anterior, este proyecto es totalmente novedoso en la Universidad Carlos III de Madrid, ya que hasta la fecha no se había realizado nada semejante y por lo tanto no existía nada similar en la misma. Por este motivo este proyecto no se puede comparar con ninguna tecnología existente dentro de la Universidad, ya que es la primera vez que se hace. Sin embargo, esto no implica que fuera de la Universidad no exista esta tecnología.

En segundo lugar, y por el motivo antes mencionado, hay que buscar esta tecnología fuera de la Universidad, es decir, ver qué existe ahora mismo en el mercado y en otras universidades relacionado con esta tecnología.

Tras un proceso de investigación, se han encontrado, por un lado, algunas empresas que ofrecen simuladores de comunicación por satélite al mercado, y por otro lado, varios artículos del IEEE [20] en los que se desarrollan también simuladores de satélites. A continuación se estudiarán las características de los simuladores de cada una de estas empresas y artículos y se compararán con el realizado en el presente proyecto.

En primer lugar, en cuanto a las empresas, cabe destacar la empresa Terma [21], que ha desarrollado varios simuladores para la Agencia Espacial Europea (ESA), aunque entre ellos destaca el simulador SIMSAT [22]. Este simulador es capaz de reproducir de una forma muy representativa el entorno real y reflejar con bastante precisión las funciones del satélite. Al igual que este proyecto, el simulador SIMSAT [21] [22] también ha sido desarrollado conforme al estándar de la ECSS [17] mencionado en el apartado del marco regulador del proyecto. La interfaz del simulador permite al usuario controlar todas las funciones y enviar secuencias de comandos para simular. También posee un planificador de tareas que permite programar comandos para que se ejecuten automáticamente y permite asimismo modificar la velocidad de la simulación. Además cuenta también con un registro (*log*) en el que se van almacenando todos los comandos que han sido simulados.

Si se compara con este proyecto, se puede ver que hay algunas características que los dos simuladores tienen en común, como que ha sido desarrollado siguiendo el estándar de la ECSS [17], que permite al usuario controlar el satélite y llevar a cabo la monitorización del mismo, que posee un planificador de tareas y que cuenta también con un registro o *log* donde se almacena toda la información de lo que se ha realizado en la simulación.

Aun así, hay una gran cantidad de funcionalidades que tiene el simulador SIMSAT [21] [22] y este proyecto no. Esto es debido a que el simulador SIMSAT ha sido desarrollado para utilizarse en misiones espaciales reales, mientras que el presente proyecto ha sido realizado con fines educativos. Por lo tanto se puede afirmar que, aunque el simulador desarrollado por Terma es mucho más complejo y completo, el simulador desarrollado en este proyecto es mucho más educativo y sencillo. A diferencia de SIMSAT [21] [22], este proyecto permite ver cómo se ha realizado la implementación de las funcionalidades y se puede comprender fácilmente qué es lo que está sucediendo en todo momento en la comunicación. Además cuenta con una interfaz sencilla para que su funcionamiento sea fácilmente comprensible.

Por otro lado, en cuanto a los artículos del IEEE [20] destacan dos artículos [23] [24] que están muy relacionados con lo que se ha llevado a cabo en este proyecto.

El primero de ellos se llama *Satellite ground station emulator: An architecture and implementation proposal* [23] y ha sido llevado a cabo por la Universidad de Manitoba (Canadá). En este artículo se explica cómo se ha llevado a cabo la implementación de un simulador de satélite con el fin de probar y entrenar una estación de tierra real.

En el artículo también se describen las funciones específicas que han desarrollado, entre las que destacan: un sistema de comunicación radio –llamado ARISS– que permite la comunicación por voz con la Estación Espacial Internacional (ISS); la adaptación a la red de estaciones de tierra GENSO [25] para permitir la recuperación de telemetría y corrección de errores; el seguimiento que puede realizar de globos sonda y sistemas lanzados por la Universidad para determinados experimentos; y la capacidad de simular errores para ver la reacción de la estación de tierra.

Si se compara con este proyecto, se pueden ver claras diferencias, como que la estación de tierra de la que habla el artículo [23] permite comunicación con voz, mientras que el de este proyecto no; o la adaptación a la red GENSO [25]. Sin embargo, la estación de tierra realizada en el presente proyecto sí que permite el seguimiento y monitorización del satélite desarrollado en este proyecto (igualmente implementado mediante software), y además también permite simular fallos desde el satélite y ver cómo reacciona la estación de tierra y los mensajes de error que genera.

Por tanto, la diferencia principal entre el presente proyecto y el del artículo [23] es que éste ha sido diseñado únicamente para un uso académico y para comunicarse con un satélite simulado mediante un transceptor NI USRP [7] – [10], mientras que el del artículo ha sido diseñado para comunicarse con satélites reales y para formar parte de la red GENSO [25].

Por otro lado, el otro artículo relacionado con el proyecto se llama *Design of a Multi-mission Satellite Ground Station for Education and Research* [24] y ha sido desarrollado por la Universidad Tecnológica de Viena (Austria). En este artículo se explican el diseño y desarrollo de una estación de tierra realizada en hardware válida para cualquier misión.

El artículo describe la realización de una estación de tierra pensada para el seguimiento de nano y pico-satélites en las bandas de frecuencia VHF (145 MHz), UHF (435 MHz) y S-band (2,3 GHz). Esta estación de tierra solamente cumple las funciones básicas de transmisión y recepción para que en un futuro se le pueda añadir cualquier funcionalidad. Aun así, la estación de tierra sí que ha sido diseñada para adaptarse a dos redes: la BRITE-Constellation [26] y la red GENSO [25].

Esta estación de tierra ha sido diseñada también con fines educativos, aunque solo permite realizar ciertas acciones como ver las señales recibidas, realizar un análisis espectral de las mismas, ver la relación señal a ruido, etc. Sin embargo, no ha sido diseñada para interpretar las señales recibidas y obtener la información, mientras que la estación de tierra diseñada en el presente proyecto es capaz de recibir e interpretar gran cantidad de información, desde parámetros de monitorización hasta imágenes.

Además, este proyecto ha sido implementado mediante software, lo que permite modificar fácilmente cualquier funcionalidad del mismo, mientras que la estación de tierra del artículo [24] ha sido implementada en hardware, lo que hace que la modificación de la misma sea complicada. También cabe destacar que, mientras que la estación de tierra del artículo [24] está sujeta a las tres frecuencias antes mencionadas, la estación de tierra del presente proyecto permite utilizar la frecuencia que se desee siempre que esté dentro del rango de frecuencias del transceptor NI USRP (de 50 MHz hasta 2,2 GHz) [9] [10].

Finalmente, con todo lo comentado anteriormente, se puede concluir que este proyecto realizado en la Universidad Carlos III de Madrid es de gran importancia ya que, aunque hay otros productos en el mercado y en otras universidades, este proyecto ofrece características que otros no ofrecen, como su orientación a la educación, la sencillez de la interfaz, o la implementación mediante software.

2.1. Marco regulador

En este apartado se describe el marco regulador en el que se ha desarrollado este proyecto.

Principalmente el proyecto está basado en dos regulaciones: el Real Decreto 1066/2001, de 28 de septiembre, por el que se aprueba el Reglamento que establece condiciones de protección del dominio público radioeléctrico, restricciones a las emisiones radioeléctricas y medidas de protección sanitaria frente a emisiones radioeléctricas [27]; y el estándar de la ECSS número ECSS-E-70-41A: *Space Engineering: Ground systems and operations – Telemetry and telecommand packet utilization*, revisado el 30 de enero de 2003 [17].

En el primer documento mencionado, el Real Decreto 1066/2001 [27], se establecen los límites de exposición del público a los campos electromagnéticos que proceden de emisiones radioeléctricas. El Real Decreto establece por tanto unas restricciones básicas y unos niveles de referencia para garantizar la salud de los ciudadanos y que todos los servicios vinculados han de cumplir, incluido este proyecto.

En el apartado 2 (restricciones básicas) del Anexo II se especifican los límites del SAR[1] (*Specific energy Absorption Rate*, Índice de Absorción Específica de Energía) para evitar un calentamiento excesivo de los tejidos debido a la radiación. En el Cuadro 1 del Real Decreto 1066/2001 [27] se muestran los límites antes mencionados: Para una frecuencia de entre 10 MHz y 10 GHz (en este proyecto se utiliza una frecuencia de 600 MHz) se establece un SAR medio de cuerpo entero de 0,08 W/kg, un SAR localizado en cabeza y tronco de 2 W/kg, y un SAR localizado en miembros de 4 W/kg.

En el apartado 3 del Anexo II se exponen también los niveles de referencia. Si el sistema respeta estos niveles de referencia se puede garantizar que respeta también las restricciones básicas antes mencionadas. En el Cuadro 2 del Real Decreto 1066/2001 [27] se establecen los niveles de referencia en función de la frecuencia utilizada. Para el caso de una frecuencia de 400 MHz hasta 2 GHz se establece un nivel de referencia de densidad de potencia equivalente a una onda plana de f/200 (W/m^2). En el caso de este proyecto, ya que la frecuencia utilizada es de 600 MHz, este valor será de 600/200 = 3 W/m^2.

Si se observan las características del NI USRP [9] [10] se puede ver que la potencia máxima emitida es de 100 mW. Por este motivo se puede afirmar que el proyecto respeta los niveles de referencia establecidos en el Real Decreto 1066/2001 [27] y por tanto cumple con las restricciones básicas que garantizan la salud de los ciudadanos.

[1] El Índice de absorción específica de energía (SAR) se define como la potencia absorbida por unidad de masa de tejido corporal. Se expresa en vatios por kilogramo (W/kg).

Por otro lado, el otro reglamento al que se ajusta este proyecto es el estándar de la ECSS número ECSS-E-70-41 [17]. Este estándar indica el formato y la forma de utilización de los paquetes de telemetría y telecomandos dedicados a la monitorización y control de los sistemas de comunicaciones por satélite.

Este estándar no incluye los comandos de telemetría y telecomandos para misiones específicas, sino que solo abarca las funcionalidades generales de monitorización y control. Aun así, estos comandos pueden ser extrapolados a otras funcionalidades específicas del sistema.

En este proyecto se han implementado un total de once servicios del sistema de comunicación por satélite. De esos servicios, ocho vienen recogidos en el estándar [17] y los tres restantes son específicos de la misión.

En primer lugar, los ocho servicios recogidos en el estándar (para más información consulte el apartado 5.1.2) se han hecho basándose en lo especificado en el mismo, aunque en alguna ocasión se ha variado algún elemento, bien para simplificar el sistema o bien para adecuarlo más al objetivo del proyecto.

Y en segundo lugar, los otros tres servicios restantes no vienen recogidos en el estándar [17] ya que son específicos del proyecto. Por este motivo estos servicios han sido propuestos por el tutor del proyecto y enteramente diseñados e implementados por los alumnos del mismo.

Para finalizar, simplemente indicar que en capítulo 5 de la memoria se expone todo lo relacionado con el estándar antes mencionado, desde los servicios que se han implementado hasta la forma de su implementación.

3. Tecnología empleada

En este capítulo de la memoria se hace una explicación teórica de la tecnología que se ha utilizado para llevar a cabo la realización de este proyecto. El objetivo de este capítulo es explicar y aclarar al lector varios conceptos y términos sobre la tecnología utilizada para que así pueda comprender de forma más sencilla la memoria.

Principalmente se hablará de dos temas en este capítulo: las comunicaciones por satélite [1] y la tecnología SDR (Software Defined Radio) [4] [5] [6]. En ambos temas se hará una breve descripción de cada uno y después se explicará cómo han sido utilizados en este proyecto.

3.1. Comunicaciones por satélite

Las comunicaciones por satélite [1] son la base de este proyecto, ya que la finalidad de este TFG es implementar un simulador de las funciones de telemetría de una estación de tierra perteneciente a una comunicación satélite-base. Como se comentó anteriormente, cuando se una este proyecto a los otros dos proyectos que implementan las funcionalidades del satélite y los telecomandos de la estación de tierra, se conseguirá realizar una simulación de una comunicación por satélite completa.

Las comunicaciones por satélite han alcanzado ya una importancia enorme en nuestra sociedad, tanto a nivel tecnológico como a nivel social. Gracias a los satélites existe una red de comunicaciones a nivel mundial que nos permite estar conectados con cualquier punto de la Tierra. Y no solo eso, ya que entre sus muchas aplicaciones, también permiten el avance de la investigación espacial al poder retransmitir información de sondas espaciales que se encuentran en los límites del Sistema Solar.

Una comunicación por satélite está formada principalmente por dos elementos: un satélite y una estación de tierra.

Un satélite de comunicaciones se define simplemente como una estación repetidora que recibe señales desde la Tierra, las procesa y las vuelve a retransmitir a la Tierra [1]. Esto se puede observar claramente en la Figura 3.1 de la siguiente página.

Figura 3.1: Satélite de comunicaciones [Figura 1.1 [1]]

Para llevar a cabo una comunicación por satélite es necesaria también la existencia de una estación de tierra que controle y se comunique con el satélite. La estación de tierra se define como un terminal de tierra fijo o móvil colocado principalmente sobre la superficie (aunque puede ser aéreo o marítimo). La estación de tierra está destinada para la comunicación con uno o varios satélites desde la Tierra [1]. Esto se puede ver a continuación en la Figura 3.2.

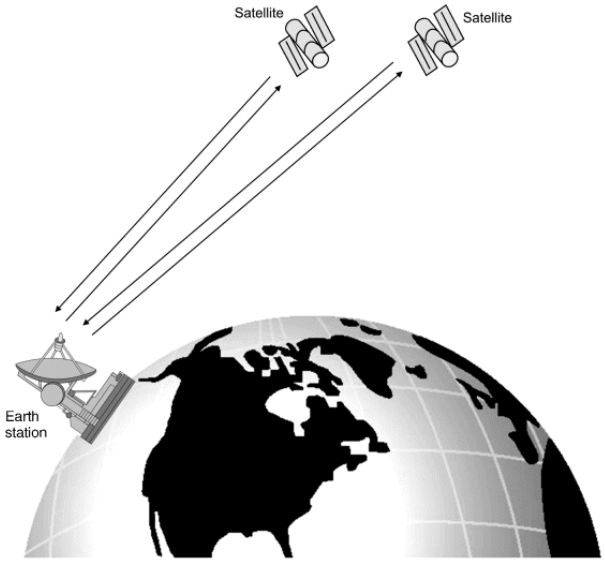


Figura 3.2: Estación de tierra comunicándose con satélites [Figura 8.1 [1]]

Para el caso de este proyecto, se va a implementar una comunicación por satélite utilizando un solo satélite y una sola estación de tierra. Aun así, como se mencionará al final de la memoria en el capítulo 8, una de las posibles mejoras a implementar en el proyecto sería ampliar el número de satélites o de estaciones de tierra.

3.1.1. Historia de las comunicaciones por satélite

La historia de las comunicaciones por satélite comenzó con un artículo de Arthur C. Clarke [28] publicado en 1945 sobre las conexiones inalámbricas, en el que por primera vez en la historia se propuso la idea de realizar una comunicación con un satélite en una órbita geoestacionaria. En este artículo explicaba cómo mediante una órbita geoestacionaria se podía conseguir que el satélite permaneciera estático para una determinada zona del planeta lo que permitía dar servicio a esa zona de forma ininterrumpida. Esto marcó el comienzo de la era de los satélites.

Entre los años 1945 y 1955 se comenzaron a realizar los primeros lanzamientos experimentales de cohetes sonda consiguiendo penetrar cada vez más en las capas más altas de la atmósfera.

En 1955, Estados Unidos y la Unión Soviética fueron los primeros países que comenzaron a elaborar planes para lanzar al espacio los primeros satélites artificiales. Fue un proceso bastante rápido, ya que en solo dos años la Unión Soviética consiguió lanzar el primer satélite, seguido de Estados Unidos que no tardó mucho en lanzar el suyo.

El primer satélite artificial enviado al espacio fue el Sputnik-1 [29], lanzado por la Unión Soviética el 4 de octubre de 1957. Este satélite orbitaba la Tierra cada 96 minutos en una órbita elíptica, y cayó a la Tierra el 4 de enero de 1958 tras 92 exitosos días en órbita.

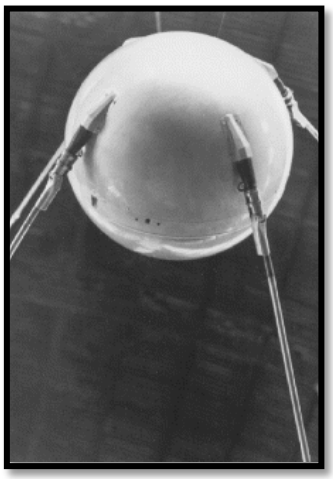


Figura 3.3: Sputnik-1 [Figura 1.7 [1]]

A estos satélites los siguieron el Sputnik-2 y el Sputnik-3. El Sputnik-2 [30] se lanzó en noviembre de 1957 y pasó a la historia por llevar dentro a la famosa perra Laika. El Sputnik-3 [31] fue lanzado en mayo de 1958 con la finalidad de estudiar la composición de la ionosfera terrestre, la radiación solar, el campo magnético y los rayos cósmicos.

En cuanto a Estados Unidos, les llevó un poco más de tiempo enviar su primer satélite artificial al espacio. Éstos finalmente lo lograron en el año 1958 con el satélite Explorer-1 [32].

Después de este lanzamiento se enviaron varios satélites artificiales más, como el Vanguard-1 [33], que fue el primero en incorporar baterías solares para recargar las baterías.

Tras estas primeras misiones y a medida que la tecnología avanzaba, los satélites fueron incorporando nuevas funcionalidades y adquiriendo nuevos cometidos. Comenzaron entonces a lanzarse los primeros satélites para comunicaciones, meteorológicos y de exploración espacial, llegando hasta nuestros días.

Desde que se lanzó el primer satélite anteriormente mencionado hasta nuestros días, se han enviado al espacio un total de 6.000 satélites. De éstos, hay 400 que siguen trayectorias interplanetarias o que han explosionado. Contando con lo anterior, se calcula que existen unos 5.600 satélites artificiales orbitando alrededor de la Tierra, aunque solo 800 de esos satélites permanecen activos [19].

3.1.2. Satélite

Un satélite [1] en términos generales es cualquier cuerpo natural o artificial que orbita alrededor de un cuerpo celeste, como un planeta. En el contexto de esta memoria, se referirá a los satélites artificiales que orbitan la Tierra.

Existen varios tipos de satélites dependiendo de la función y cometido del mismo: En primer lugar los satélites de comunicaciones, que son simples repetidores que reciben la información desde la Tierra y la vuelven a retransmitir. En segundo lugar los satélites de observación, cuyo cometido es obtener datos y fotografías de aquello a lo que están observando para llevar a cabo avances en la exploración espacial. Y en tercer lugar los satélites meteorológicos, que toman datos de distintos parámetros atmosféricos.

En este proyecto, el satélite que se ha simulado es un satélite de observación, ya que su objetivo es enviar los datos obtenidos desde los sensores, como pueden ser los datos de temperatura o las imágenes tomadas con la cámara. El satélite simulado sería por tanto parecido al que se muestra en la Figura 3.4.

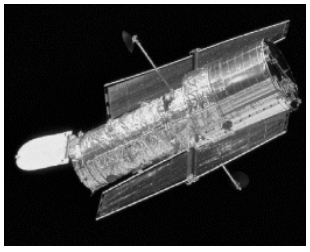


Figura 3.4: Satélite de observación [Figura 1.5 [1]]

3.1.3. Estación de tierra

La estación de tierra [1] es un terminal situado sobre la superficie de la Tierra y cuyo objetivo es estar en comunicación con el satélite. En la mayoría de implementaciones las estaciones de tierra realizan transmisión y recepción de datos con los satélites, aunque existen casos especiales en los que la estación de tierra solo realiza una de las dos funciones (la transmisión o la recepción de datos).

Toda estación de tierra cuenta con un sistema de transmisión y/o un sistema de recepción. El sistema de transmisión se lleva a cabo implementando comandos de telecomandos, mientras que el sistema de recepción se lleva a cabo mediante la implementación de comandos de telemetría. En este TFG se ha realizado la implementación de los comandos de telemetría del sistema de recepción de la estación de tierra.

En el proyecto conjunto la estación de tierra que se simula posee tanto el sistema de transmisión como el de recepción, aunque cada parte sea implementada por un alumno. En conjunto, la estación de tierra permite tanto enviar datos al satélite como recibirlos. La estación de tierra simulada sería por tanto parecida a la que se muestra a continuación en la Figura 3.5.

Figura 3.5: Estación de tierra de transmisión y recepción [Figura 8.4 [1]]

3.2. SDR (Software Defined Radio)

Debido al avance casi exponencial de la tecnología en los últimos años, se han producido constantes mejoras en los sistemas de comunicaciones. Pero, si cada vez que se produce uno de estos cambios hubiera que cambiar por completo el sistema radio implementado sería un proceso muy laborioso y costoso. Es aquí donde entra en juego la tecnología Software Defined Radio [4] [5] [6].

Existen varias definiciones de lo que es la tecnología SDR, pero la más sencilla y a la vez completa es la siguiente: “Radio en la que algunas o todas las funciones de las capas físicas están definidas por software” [6].

Los beneficios de esta tecnología son inmensos [4] [5] [6], aunque cabe destacar los siguientes:

En primer lugar, para los fabricantes de sistemas de radio la existencia de la tecnología SDR permite implementar productos que utilizan una plataforma de arquitectura común, lo que permite introducir nuevos productos al mercado con mayor rapidez. Además, las plataformas SDR permiten reutilizar el mismo hardware, lo que reduce enormemente los costes.

En segundo lugar, para los proveedores (operadores) supone una mejora en cuanto a que esta tecnología es muy flexible, lo que les permite desarrollar e introducir rápidamente y de forma sencilla nuevos servicios sin realizar inversiones importantes de capital.

Finalmente, en cuanto a los clientes, el uso de esta tecnología supone un ahorro para ellos ya que también supone un ahorro para los operadores. Además, esta tecnología también permite proporcionar a los clientes un servicio más flexible y adaptado.

En el año 2011, el *Wireless Innovation Forum* [6] encargó a *Mobile Experts LLC* [34] la realización de un estudio de mercado para evaluar el nivel de aceptación de esta tecnología en el mercado. En este estudio se vio que la tecnología SDR había sido adoptada con gran éxito, ya que, por ejemplo, alrededor del 93% de la infraestructura móvil y prácticamente todos los sistemas militares de comunicaciones utilizaban ya la tecnología SDR.

Además, en el siguiente gráfico de la Figura 3.6 se puede observar una comparativa del número de estaciones bases que fueron lanzadas con tecnología definida por software (SDR) frente a las controladas por software.

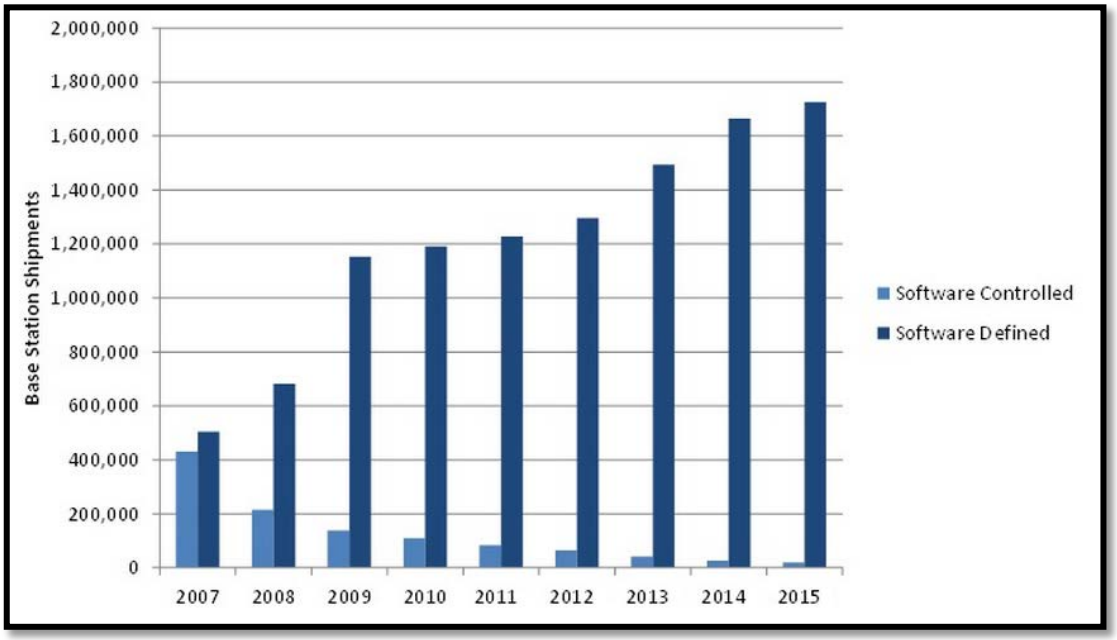


Figura 3.6: Gráfico comparativo [35]

En el gráfico anterior se puede ver la gran evolución mencionada anteriormente que ha tenido esta tecnología.

En cuanto a este proyecto, la tecnología SDR [4] [5] [6] [35] es un concepto clave ya que absolutamente todo ha sido implementado mediante esta tecnología. Como se comentará más adelante en el capítulo 4 de la memoria, la implementación de este proyecto se ha llevado a cabo utilizando unos transceptores NI USRP [7] con tecnología SDR y la plataforma de programación en flujo LabVIEW [14] [15] [16] que permite programarlos.

4. Entorno de trabajo

En este capítulo se describen las herramientas (tanto de hardware como de software) que se han utilizado para llevar a cabo la implementación del proyecto. En cuanto a hardware se han utilizado transceptores NI USRP 2920 [7] [8] [9] [10], y en cuanto a software se ha utilizado el programa LabVIEW [14] [15] [16] para programar los transceptores.

Además de esto, también ha sido necesario para la implementación del proyecto un ordenador con puerto Gigabit Ethernet [36] para poder conectar el transceptor al ordenador.

A continuación se explican brevemente las dos herramientas antes mencionadas.

4.1. NI USRP 2920

Un NI USRP (Universal Software Radio Peripheral) [7] [8] [9] [10] es un transceptor que puede transmitir y recibir señales de radiofrecuencia dentro de un ancho de banda determinado. Son muy utilizados tanto en ámbitos educacionales como de investigación en radio frecuencia debido a su gran capacidad de flexibilidad y adaptación. Se puede ver cómo es el transceptor y cómo es su panel frontal en las Figuras 4.1 y 4.2 respectivamente.

Figura 4.1: NI USRP 2920 [8]

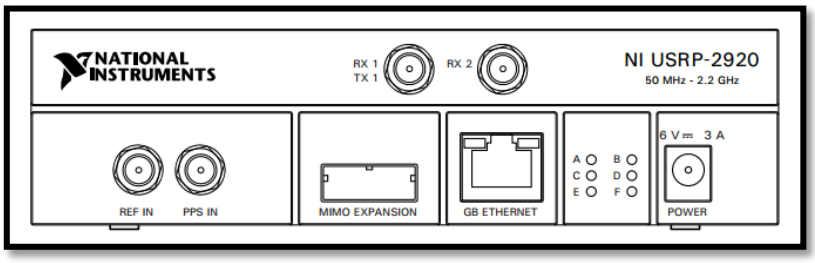


Figura 4.2: Panel frontal de un NI USRP 2920 [Figura 1 [9]]

Las principales características del transceptor NI USRP 2920 [8] [9] [10] son las siguientes:

Pueden trabajar en un rango de frecuencias desde los 50 MHz hasta los 2,2 GHz. Para este proyecto las frecuencias de transmisión y recepción utilizadas son de 650 MHz para el enlace de subida (estación de tierra -> satélite), y de 600 MHz para el enlace de bajada (satélite -> estación de tierra). Se han utilizado estas frecuencias ya que pertenecen a un rango de frecuencias poco utilizado y reservado para la televisión [11] [12], por lo que se reduce el problema de las interferencias.

En cuanto a la transmisión, pueden transmitir con una potencia de entre 17 dBm a 20 dBm para un rango de frecuencias de 50 MHz a 1,2 GHz; y de 15 dBm a 18 dBm para un rango de frecuencias de 1,2 GHz a 2,2 GHz. Además tiene un rango de ganancia de 0 dB a 31 dB para la transmisión y de 0 dB a 31,5 dB para la recepción.

En cuanto a la recepción, tiene un ancho de banda instantáneo de 40 MHz, y se ha utilizado con la antena colocada en el puerto RX1 (véase Figura 4.2).

A continuación se muestra también en la Figura 4.3 el diagrama de bloques del sistema hardware.

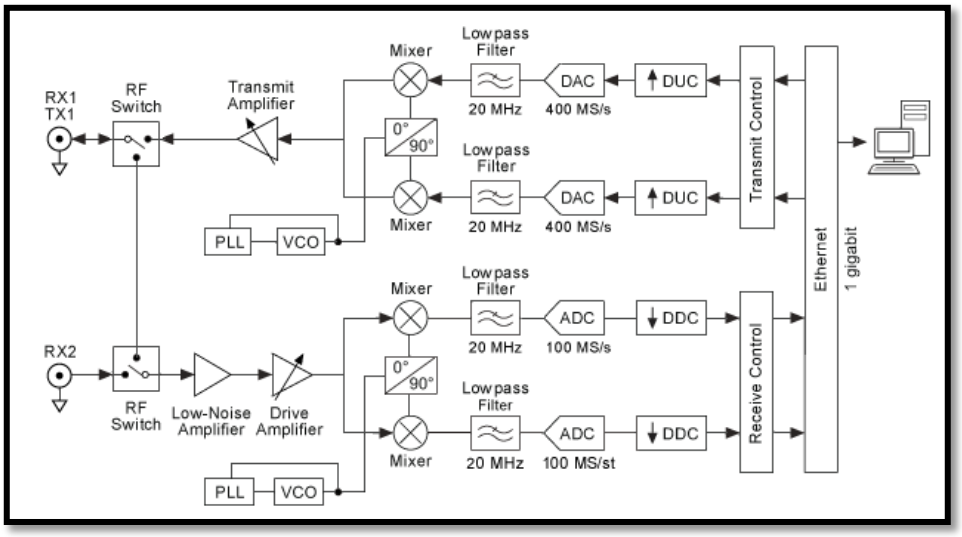


Figura 4.3: Diagrama de bloques del sistema hardware [Figura 2 [7]]

Como se puede observar en la figura anterior, la recepción comienza con el terminal de recepción RX. Es un terminal analógico y muy sensible capaz de recibir señales muy pequeñas y digitalizarlas a una señal en fase (I) y cuadratura (Q). Después de esto se realiza la conversión de analógico a digital (ADC) de la señal y se adapta para que pueda ser interpretada por el ordenador mediante un DDC (Digital DownConversion) para su posterior procesamiento.

En cuanto a la transmisión, el proceso es el contrario. La señal sale del ordenador, pasa a través del DUC (Digital UpConversion) que prepara las señales

para el conversor de digital a analógico (DAC). Una vez transformada la señal a analógico, se mezclan la señal en fase (I) y en cuadratura (Q), y finalmente se amplifica y se transmite por el terminal de transmisión TX.

4.2. LabVIEW

En este apartado se explica brevemente el entorno de programación LabVIEW [14] [15] [16] que ha sido utilizado para programar los transceptores NI USRP [7] [8] [9] [10].

LabVIEW (Laboratory Virtual Instrument Engineering Workbench) es un lenguaje y a la vez un entorno de programación gráfica desarrollado por la empresa National Instruments [13]. Esta empresa lanzó el primer programa LabVIEW 1.0 en el año 1986 [15], y ha ido lanzando al mercado varias versiones nuevas del producto hasta llegar a LabVIEW 2015, que es la última que existe hasta el momento. Sin embargo, para este proyecto se ha utilizado la versión de 2010 de LabVIEW por problemas logísticos de la Universidad.

El entorno del programa LabVIEW [14] [15] [16] es fácil de comprender y utilizar. El programa cuenta con dos ventanas principales que están completamente relacionadas entre sí: el panel frontal y el diagrama de bloques.

El panel frontal es la interfaz con la que se encuentra el usuario. En el panel frontal se encuentran todas las entradas y salidas del programa representadas de forma gráfica. Éstas pueden ser de muchos tipos: botones, indicadores numéricos o de texto, indicadores luminosos (LEDs), gráficas, imágenes, etc. Todo esto permite al usuario controlar el programa y también ver los resultados del mismo.

Por otro lado, el diagrama de bloques es donde se realiza la programación del programa. Todo programa implementado en un diagrama de bloques constará de los siguientes elementos: controles, que representan las entradas de los datos; funciones, que realizan operaciones con esos datos; e indicadores, que son las salidas de los datos. Todos estos elementos se conectan mediante cables por los que circulan los datos del programa.

Por cada elemento de entrada o salida que se añade en el panel frontal aparece su respectivo control o indicador en el diagrama de bloques. A continuación se puede observar en la Figura 4.4 un ejemplo de panel frontal con varios elementos, tanto de entrada como de salida, y su respectivo diagrama de bloques donde aparecen los controles e indicadores de esos elementos.

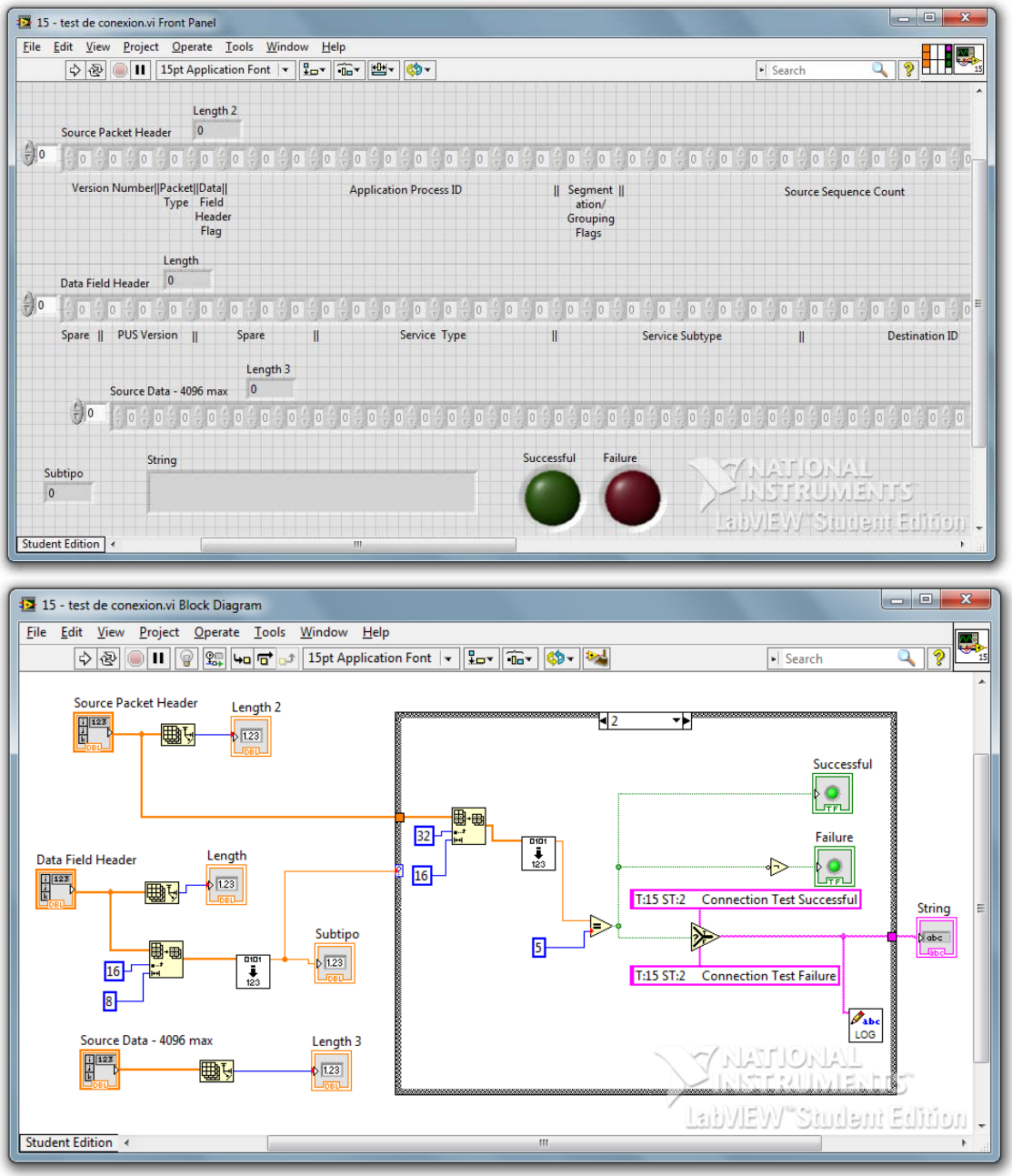


Figura 4.4: Panel frontal y diagrama de bloques

Un programa en LabVIEW [14] [15] [16] se denomina “instrumento virtual” o VI, debido a que su apariencia es similar a la de instrumentos físicos como osciloscopios o multímetros.

En cuanto a la forma de programación, el lenguaje que utiliza LabVIEW [14] [15] [16] también se conoce como lenguaje G, que tiene una ejecución basada en flujo de datos (dataflow). Esto quiere decir que una función solo se podrá ejecutar cuando tenga disponibles todos los datos que necesita como entradas. De esta forma se pueden ejecutar varios procesos de forma paralela, al igual que en el lenguaje VHDL [37].

A continuación se muestra un ejemplo de programación en LabVIEW. Si, por ejemplo, quisiéramos realizar la operación $c = (a + 5) \times b$, el diagrama de bloques sería de la siguiente manera:

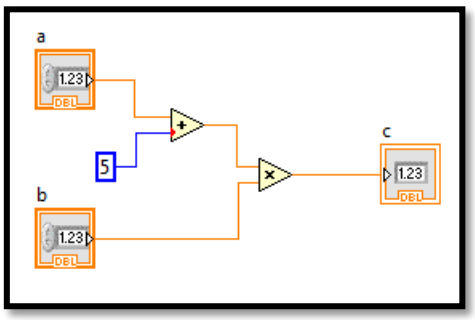


Figura 4.5: Ejemplo de programación

Como se puede ver en la Figura 4.5, al tratarse de programación en flujo, hasta que la función de suma no devuelva un valor la función de multiplicación no podrá ejecutarse y obtener el resultado final.

5. Diseño y desarrollo

En el presente capítulo se detalla cómo se ha llevado a cabo la implementación de las funciones de telemetría de la estación de tierra, cumpliendo con los objetivos propuestos al comienzo de la memoria.

El capítulo comienza con una explicación teórica (en la que se exponen brevemente los servicios a implementar) y con una visión general del programa llevado a cabo con la plataforma LabVIEW [14] [15] [16]. Tras esto, se explican de forma más detallada los módulos realizados para llevar a cabo la implementación. Esta última parte constará de una descripción más profunda del programa ya que, como se mencionó al comienzo de la memoria, este proyecto está orientado a que futuras generaciones lo mejoren y amplíen, por lo que debe quedar clara constancia de lo realizado de forma que sea fácilmente comprensible para el siguiente continuador del proyecto.

5.1. Sistema implementado

En este apartado se ofrece una visión general del sistema que se ha implementado y de cómo se ha hecho.

Antes de comenzar, se ha de puntualizar que todo este proyecto se ha realizado en base y conforme al estándar de la ECSS número ECSS-E-70-41A [17]: *Space Engineering: Ground systems and operations – Telemetry and telecommand packet utilization*, revisado el 30 de enero de 2003.

Como se comentó anteriormente en el apartado 2.1, en este estándar se especifica cómo se ha de implementar la configuración de una comunicación por satélite. Y de esa forma es como se ha realizado el proyecto, salvo alguna pequeña excepción que se comentará más adelante.

En el estándar también está recogido cómo han de ser los paquetes de la función de telemetría y los servicios que han de llevarse a cabo. Esto está expuesto de forma esquemática en sendos Anexos A y B, y se desarrolla de forma más detallada a continuación.

5.1.1. Estructura de los paquetes de telemetría

Toda la información relativa a la estructura de los paquetes de telemetría viene recogida en el capítulo 5.4 del estándar de la ECSS [17] mencionado anteriormente. A continuación se expone lo más relevante del estándar.

Para empezar, el esquema de la estructura de los paquetes de telemetría se muestra a continuación en la Figura 5.1. Después, se hace una breve descripción de cada uno de los campos del paquete.

Packet Header (48 Bits)							Packet Data Field (Variable)		
Packet ID				Packet Sequence Control		Packet Length	Data Field Header	Source Data	Packet Error Control
Version Number (=0)	Type (=0)	Data Field Header Flag	Applica-tion Process ID	Grouping Flags	Source Sequence Count				
3	1	1	11	2	14				
16				16		16	32	Variable	16

Figura 5.1: Campos del paquete de Telemetría [Figura 4[17]]

Como se puede observar en la figura 5.1, el paquete de telemetría está formado por la cabecera (*Packet Header*), que está formada por 48 bits; y el campo de datos, que tiene una longitud variable hasta un máximo de 680 bits. A continuación se describe brevemente cada uno de los campos que aparecen en la figura.

Packet Header

Es la cabecera del paquete y contiene la información necesaria para caracterizar e identificar el paquete. Está formado por el "Identificador de paquete" (*Packet ID*), el "Control de la secuencia de paquetes" (*Packet Sequence Control*), y la "Longitud del paquete" (*Packet Length*).

Packet ID (16 Bits)

Version Number:

Indica la versión del tipo de paquete de telemetría. Este campo está diseñado por si en un futuro se realizan nuevas versiones del paquete. Por el momento este campo tomará siempre el valor '000' ya que estamos ante la primera versión del paquete.

Type:

Este bit permite distinguir entre paquetes de telecomando y paquetes de telemetría. Para los paquetes de telemetría tomará el valor '0'.

Data Field Header Flag:

Este bit indica la presencia o ausencia del campo *Data Field Header*. Todos los paquetes de telemetría cuentan con este campo, luego este bit tomará siempre el valor '1'.

Application Process ID (APID):

Se corresponde de manera inequívoca con un proceso de aplicación de a bordo que es el que genera el paquete.

Packet Sequence Control (16 Bits)

Grouping Flags:

Este campo se utiliza cuando se envían varios paquetes de telemetría de un mismo proceso de aplicación. Es decir, cuando la información es demasiado grande como para enviarla en un único paquete y se ha de dividir en varios. La interpretación de estos dos bits es la siguiente:

- '01' significa que es el primer paquete de un grupo de paquetes.
- '00' indica que es un paquete intermedio.
- '10' significa que es el último paquete de un grupo de paquetes.
- '11' indica que es un paquete único y que no ha sido dividido en más paquetes.

Source Sequence Count:

Este campo se utiliza para representar el número actual del contador de la secuencia y es incrementado de uno en uno por cada paquete que es enviado. Este campo sirve para saber si ha habido pérdida de paquetes y también para saber el orden de los mismos.

Packet Length (16 Bits)

Este campo contiene la longitud del campo *Packet Data Field.* Esta longitud se transforma a octetos y se le resta 1. El resultado de la operación es la información que se incluye en este campo.

Packet Length = Número de octetos de *Packet Data Field* – 1

Packet Data Field

Este campo del paquete es el que contiene la información y los datos que se envían desde el satélite. Está formado por la "Cabecera del campo de datos" (*Data Field Header*), los datos (*Source Data*) y el "Control de error de paquetes" (*Packet Error Control*).

Data Field Header (32 Bits)

La cabecera del campo de datos contiene la información relativa al tipo y subtipo de servicio del que se está enviando la información, así como del identificador del destinatario. Los campos de la cabecera del campo de datos se muestran de forma esquemática a continuación en la Figura 5.2. Después se hace una breve descripción de cada uno de los campos antes mencionados.

Spare	TM Source Packet PUS Version Number	Spare	Service Type	Service Subtype	Destination ID
1 bit	3 bits	4 bits	8 bits	8 bits	8 bits

Figura 5.2: Campos de la cabecera del campo de datos

Spare:

Este bit de relleno siempre toma el valor '0'. Se utiliza para que el número final de bits sea un múltiplo de 8 y para guardar simetría con la cabecera del campo de datos de los paquetes de telecomandos.

TM Source Packet PUS Version Number:

Este campo indica el número de versión del paquete. Está pensado por si se introducen mejoras en el futuro y se cambian los campos del paquete. Por el momento este campo tomará siempre el valor 1 ('001') ya que corresponde a la primera versión.

Spare:

Estos cuatro bits se introducen para completar el octeto y hacer que la cabecera del campo de datos tenga un número de bits que sea múltiplo de 8. Estos cuatro bits tomarán el valor 0.

Service Type:

Indica el tipo de servicio al que el paquete de telemetría se refiere. Los servicios que se han implementado en este proyecto se pueden ver de forma esquemática en el Anexo B y de forma detallada en el apartado 5.1.2.

Service Subtype:

Conjuntamente con el campo anterior, este campo identifica el subtipo de servicio al que pertenece el paquete de telemetría. Conjuntamente con el tipo y subtipo de servicio, que han de ser únicos e inequívocos, se determina la acción a realizar por la estación de tierra. Los servicios que se han implementado en

este proyecto se pueden ver de forma esquemática en el Anexo B y de forma detallada en el apartado 5.1.2.

Destination ID:

Este campo identifica al destinatario del paquete de telemetría. Como en este proyecto solo hay un destinatario (la estación de tierra), se deja este campo para futuras mejoras en las que se incluyan más de un destinatario.

Source Data (longitud variable)

Este campo está formado por los datos relativos al servicio correspondiente que se envían desde el satélite a la estación de tierra.

Packet Error Control (16 Bits)

Este campo de control de errores está diseñado para detectar errores en los paquetes y verificar la integridad del paquete. Esta funcionalidad no se ha podido diseñar en este proyecto por falta de tiempo, pero sí que se han tenido en cuenta y se han reservado estos 16 bits a la hora de implementar el sistema. Por lo tanto el control de errores en los paquetes queda como futura mejora para el proyecto.

5.1.2. Servicios implementados

En este apartado se expone una breve descripción de los servicios que se han implementado en este proyecto. En esta sección se hablará de los tipos de servicio que se han llevado a cabo (11 en total). En el apartado 5.2 se hará una descripción detallada de cómo se ha realizado cada uno de los subtipos de cada uno de los servicios de telemetría implementados.

Además, el Anexo B consta de una tabla donde se encuentran todos los tipos y subtipos de servicios tanto de telemetría (implementados entre este proyecto), como los de telecomandos (implementados en los proyectos realizados por los otros dos miembros del proyecto), que permite al lector tener una mejor visión general del proyecto.

Antes de comenzar se ha de aclarar que la mayor parte de los servicios implementados, ocho para ser más exactos, son servicios básicos que vienen recogidos en el estándar de la ECSS [17]. Mientras que los tres restantes son servicios específicos (gestión del sensor de imágenes y de temperatura, y reinicio del sistema) que no vienen recogidos en el estándar y que han sido totalmente ideados por los miembros del proyecto y por el tutor del mismo.

Servicio 1: Verificación de Telecomando

Este primer servicio se corresponde también con el servicio 1 del estándar [17], y permite a la estación de tierra verificar el telecomando enviado anteriormente por la misma, sabiendo en qué estado se encuentra. Esto quiere decir que permite saber si el telecomando ha sido aceptado o no por el satélite y si se ha ejecutado correctamente o no.

Este servicio está por lo tanto unido a todos los demás servicios implementados, ya que cada vez que se envíe un telecomando el satélite responderá con este servicio informando del estado de aceptación y ejecución del telecomando.

Además, en caso de que se produzca un error en la aceptación o ejecución del telecomando, el satélite enviará un informe especificando la causa del error.

Servicio 3: Housekeeping y Diagnóstico de Datos

Este servicio se corresponde también con el servicio 3 del estándar [17] (capítulo 8), y es el encargado de reportar a la estación de tierra toda la información relevante que no es reportada por otros servicios como el de Eventos (servicio nº5). En nuestro caso, se enviará con este servicio toda la información de housekeeping relativa al microprocesador y a los sensores de imagen y temperatura.

El servicio de housekeeping (gestión interna) enviará periódicamente informes del estado de los diferentes componentes del satélite. En nuestro caso, como se ha mencionado en el párrafo anterior, se controlará el estado del microprocesador y de los sensores. El estándar [17] se refiere a cada uno de estos componentes como estructura, y son identificados mediante un "identificador de estructura" –en inglés: *structure identification* (SID)–, que debe ser único e inequívoco. Dentro de cada estructura se enviarán informes de distintos aspectos del componente, como la temperatura o la frecuencia de funcionamiento.

Además de esto, el servicio permite también añadir o eliminar, y activar o desactivar parámetros de housekeeping. Asimismo permite modificar el tiempo de generación de los informes y de adquisición de los datos.

Servicio 5: Eventos

Este servicio se corresponde con el servicio número 5 del estándar [17], cuya información viene recogida en el capítulo 10 del mismo. Se encarga de generar informes sobre el funcionamiento de los componentes del satélite. La diferencia con el servicio anterior de housekeeping es que estos informes no se envían de forma periódica, sino que solo se envían cuando se detecta algún fallo o anomalía en el funcionamiento o cuando se realiza alguna acción a bordo de forma autónoma.

Es decir, este servicio se encarga de la generación de informes para notificar a la estación de tierra cualquier evento significante ocurrido en el satélite. Existen cuatro niveles distintos de informe:

- Normal report → Progreso normal.
- Error/anomaly report - low severity → Error de severidad baja.
- Error/anomaly report - medium severity → Error de severidad media.
- Error/anomaly report - high severity → Error de severidad alta.

Debe permitir también activar y desactivar la generación y notificación de eventos.

Servicio 6: Gestión de Memoria

Este servicio se corresponde con el servicio número 6 del estándar [17] y viene desarrollado en el capítulo 11 del mismo. Es el encargado de gestionar y manejar las distintas áreas de memoria que tiene el satélite a bordo. Permite cargar, vaciar y verificar el contenido de las diferentes áreas de la memoria del satélite.

La forma de identificar los bloques de memoria será mediante un identificador absoluto *Start Address,* que señala dónde comienza el bloque de memoria; y un campo *Length* que determina la longitud del bloque de memoria en cuestión.

Servicio 9: Gestión del Tiempo

Este servicio viene recogido en el capítulo 13 del estándar [17] y se corresponde con el servicio número 9 del mismo.

El servicio de gestión del tiempo se encarga de manejar y controlar el tiempo de referencia del satélite que éste tiene a bordo. Con este servicio se puede solicitar al satélite un informe con el tiempo de referencia que posee y, además, también permite modificarlo.

En el apartado 5.2 se desarrolla de forma más detallada cómo se han codificado la fecha y hora del satélite para poder transmitirlas y modificarlas.

Servicio 11: Gestión del Planificador de Telecomandos

El presente servicio coincide con el servicio 11 del estándar [17] y está detallado en el capítulo 14 del mismo.

Permite ordenar la ejecución de los telecomandos que han sido cargados previamente en el satélite. Para llevar esto a cabo, el satélite dispone de un planificador a bordo en el que están recogidos todos lo telecomandos que debe ejecutar y la hora a la que deben ser ejecutados.

El usuario, desde la estación de tierra, debe poder además ser capaz de activar o desactivar telecomandos, así como de eliminar o añadir nuevos

telecomandos. También debe ser capaz de solicitar un informe con los telecomandos que están programados y la hora a la que están programados.

Los telecomandos del planificador tendrán un identificador (Schedule ID) único e inequívoco que permita distinguir el telecomando y constarán de un tiempo absoluto (fecha y hora) que indica cuándo han de ser ejecutados. Esto se verá de forma más detallada en el apartado 5.2.

Servicio 12: Monitorización de a Bordo

Este servicio viene recogido en el capítulo 15 del estándar [17] y se corresponde con el servicio 12 del mismo.

El servicio de monitorización de a bordo proporciona la capacidad de monitorizar los parámetros de a bordo en relación a las comprobaciones definidas desde la estación de tierra. Reporta la verificación de cualquier cambio producido en el satélite. Para lograr esto, el servicio mantiene una lista de monitorización con la información de los parámetros que se están monitorizando.

El usuario desde la estación de tierra tiene que ser capaz de activar o desactivar los parámetros de monitorización, añadir o eliminar parámetros de monitorización, solicitar informes con los parámetros que están siendo monitorizados, y otras funciones que se detallarán en el apartado 5.2.

Cada uno de los parámetros a monitorizar debe tener un identificador que lo distinga de los demás, lo que en el estándar [17] llama *structure identifier* (SID). Además, cada estructura puede tener uno o varios parámetros a monitorizar. Tanto las estructuras como los parámetros tienen que tener un identificador unívoco que permita diferenciarlos del resto.

Servicio 15: Test de Conexión

Este servicio se corresponde con el servicio 17 del estándar [17] y viene recogido en el capítulo número 19.

El servicio de test de conexión permite activar las funciones de test implementadas en el satélite y reportar los resultados de esos test. El objetivo de este servicio es comprobar que existe comunicación entre el satélite y la estación de tierra. Para ello desde la estación de tierra se enviará un paquete vacío que deberá ser respondido con otro paquete vacío por el satélite si éste lo ha recibido correctamente.

Este servicio es de gran utilidad ya que permite saber en cualquier momento si existe o no comunicación entre el satélite y la estación de tierra.

Servicio 32: Instrumento de Sensor de Temperatura

Este servicio es un servicio específico y ha sido ideado por nuestro tutor y por el equipo del proyecto, y por lo tanto no viene recogido en el estándar [17].

Se encarga de gestionar todo lo relativo al sensor de temperatura con el que se supone que cuenta el satélite. Este sensor tomará datos de la temperatura de forma periódica y los enviará a la estación de tierra.

Además de esto, este servicio también permite al usuario encender o apagar el instrumento, cambiar el intervalo de adquisición del sensor, solicitar en cualquier momento un informe con los datos de los que dispone, y solicitar un test del instrumento para conocer el estado del sensor.

Servicio 64: Instrumento de Imágenes

Al igual que el servicio anterior, este servicio tampoco viene recogido en el estándar [17] ya que es un servicio específico. También ha sido ideado por nuestro tutor y por los miembros del proyecto.

Gestiona todo lo relativo al sensor de imágenes con el que se supone que contará el satélite y que le permitirá tomar fotografías en el espacio. El satélite tomará datos (fotografías) del sensor de imágenes y los enviará de forma periódica a la estación de tierra.

De igual modo que en el anterior servicio, éste ha de permitir al usuario encender y apagar el instrumento, cambiar el intervalo de adquisición del sensor, solicitar en un momento dado que envíe los datos de los que dispone, y solicitar un test del instrumento para saber si el sensor se encuentra operativo o no.

Servicio 128: Reboot

Este servicio también ha sido ideado por nuestro tutor y por el equipo del proyecto, y por lo tanto no viene recogido en el estándar [17].

La función de este servicio es la de reiniciar el sistema de a bordo por completo, volviendo a poner todos los valores a los valores iniciales. Sin embargo, solo cuenta con un comando de telecomando (que es el que da la orden de llevar a cabo el reinicio) y no cuenta con ningún comando de telemetría. Por este motivo en esta memoria no aparecerá cómo se ha implementado este servicio, ya que el presente proyecto se ciñe solamente a la parte de telemetría.

5.2. Módulos

En este apartado se analizan los módulos más importantes que se han llevado a cabo para implementar los servicios de telemetría de la estación de tierra. Se detallará cómo se han hecho cada uno de los tipos y subtipos de servicios y se comentarán los programas implementados con LabVIEW [14] [15] [16] que se han realizado.

Este proyecto está realizado con 12 módulos principales. Uno de ellos (*Estación de Tierra.vi*) es el encargado de la recepción de paquetes, otro (*Elegir servicio.vi*) se encarga de elegir el servicio que se ha de ejecutar, y los diez restantes son los que realizan las funcionalidades de los correspondientes servicios implementados. A continuación se muestra una lista de los doce módulos antes mencionados con una breve descripción del cometido de cada uno.

- *Estación de Tierra.vi*: Éste es el módulo principal del programa. Es el encargado de la recepción de los paquetes, aunque no los interpreta. Está formado por un bucle que hace que se esté ejecutando continuamente y todos los demás módulos están supeditados a él, es decir, jerárquicamente éste es el módulo principal. La información que le llega desde el transceptor USRP la transmite directamente al siguiente módulo (*Elegir servicio.*vi) que se encarga de interpretarla. Además, este módulo es también el encargado de mostrar los resultados del programa mediante texto e indicadores. Esta información será transmitida desde el módulo mencionado anteriormente (véase Figura 5.3).

- *Elegir servicio.vi*: Este módulo recibe la información enviada por el módulo anterior y lee la cabecera del paquete (*Packet Header*). De este modo interpreta la información contenida en la cabecera, es decir, comprueba los campos que se explicaron en el apartado 5.1. Además de esto, este módulo también lee la cabecera del campo de datos (*Data Field Header*) y determina de qué tipo de servicio es el paquete que se ha recibido. En función de este parámetro, se ejecuta el módulo correspondiente a ese tipo de servicio y le transmite toda la información. Además también transmite al módulo superior (*Estación de tierra.vi*) los resultados que le llegan desde los módulos inferiores (véase Figura 5.3).

- *1 - verificación.vi*: Éste es el primero del grupo de módulos que llevan a cabo la implementación de un servicio. El módulo lee la cabecera de los datos (*Data Field Header*) para saber el subtipo de servicio al que pertenece el paquete recibido. Este servicio tiene cuatro subtipos de servicios de telemetría que se detallarán más adelante. El programa actuará de una determinada manera en función del subtipo de servicio del paquete y transmitirá los resultados al módulo superior (*Elegir servicio.vi*) (véase Figura 5.3).

- *3 - housekeeping y diagnóstico de datos.vi*: Al igual que el anterior, este módulo interpreta la cabecera de los datos para saber a qué subtipo de servicio pertenece y actúa en consecuencia. Este servicio consta de dos subtipos de telemetría. De igual modo, este módulo también transmite los resultados al módulo superior (véase Figura 5.3).

- *5 - eventos.vi*: Este módulo es el encargado de gestionar los eventos y determina el subtipo del servicio del paquete interpretando la

cabecera de los datos (*Data Field Header*). Este servicio cuenta con seis subtipos de telemetría, los cuales se explicarán con más detalle más adelante. Este módulo también está supeditado al módulo superior (*Elegir servicio.vi*), que es el que le transmite la información; y éste le transmite de vuelta los resultados para que los muestre (véase Figura 5.3).

- *6 - gestión memoria.vi*: El siguiente módulo se encarga de todo lo relativo a la gestión de memoria. Este servicio consta de dos subtipos de telemetría y, al igual que los anteriores, decide a cuál de ellos pertenece el paquete interpretando la información de la cabecera de los datos. Tras ejecutarse devuelve los resultados al módulo superior (véase Figura 5.3).

- *9 - gestión del tiempo.vi*: Este módulo actúa de forma similar a los anteriores, ya que obtiene la información relativa al subtipo de la cabecera y devuelve los resultados obtenidos (véase Figura 5.3). La diferencia es que este módulo es el encargado de gestionar todas las funciones relativas al tiempo. Cuenta con un único comando de telemetría.

- *11 - gestión del planificador de TC.vi*: Este servicio implementado en este módulo cuenta solamente con un comando de telemetría. Éste es el encargado de interpretar y mostrar una lista detallada de todos los telecomandos que se han programado. De todas formas, aunque solo hay un subtipo de servicio de telemetría, el programa comprueba leyendo la cabecera que el subtipo de servicio sea el correcto.

- *12 - monitorización de a bordo.vi*: Este servicio es estructuralmente muy parecido al anterior, ya que solo dispone de un comando de telemetría y éste además es el encargado de interpretar y mostrar la lista detallada de todos los parámetros que están siendo monitorizados. Al igual que en el anterior, este módulo también comprueba que el subtipo que aparece en la cabecera de los datos es el correcto.

- *15 - test de conexión.vi*: Este servicio, que es el encargado de comprobar que existe comunicación entre el satélite y la estación de tierra, también cuenta con un único comando de telemetría. Éste consiste en interpretar la respuesta que envía el satélite como señal de que la comunicación existe y como contestación al telecomando en el que se solicita el test. Este módulo también comprueba que el subtipo de la cabecera es el correcto.

- *32 - instrumento sensor temperatura.vi*: Este módulo es el encargado de gestionar todo lo relativo a los datos de temperatura, y determina el subtipo del servicio del paquete interpretando la cabecera de los datos (*Data Field Header*). Este servicio consta de tres subtipos de telemetría. Al igual que todos los anteriores, este módulo también

recibe la información desde el módulo superior (*Elegir servicio.vi*), al que está subordinado; y éste le transmite de vuelta los resultados para que los muestre y represente de forma gráfica (véase Figura 5.3).

- *64 - instrumento imágenes.vi*: Este servicio es muy parecido al anterior, solo que en vez de tratar con datos de temperatura trata con datos de imagen (fotografías). El servicio consta también de tres subtipos de telemetría, y determina a cuál pertenece el paquete interpretando la cabecera de los datos. Al igual que los demás, este módulo también recibe la información del módulo superior y devuelve los resultados a éste tras ejecutarse (véase Figura 5.3).

A continuación se muestra en la Figura 5.3 un esquema en el que se refleja la jerarquía de los módulos mencionados anteriormente y que conforman el programa.

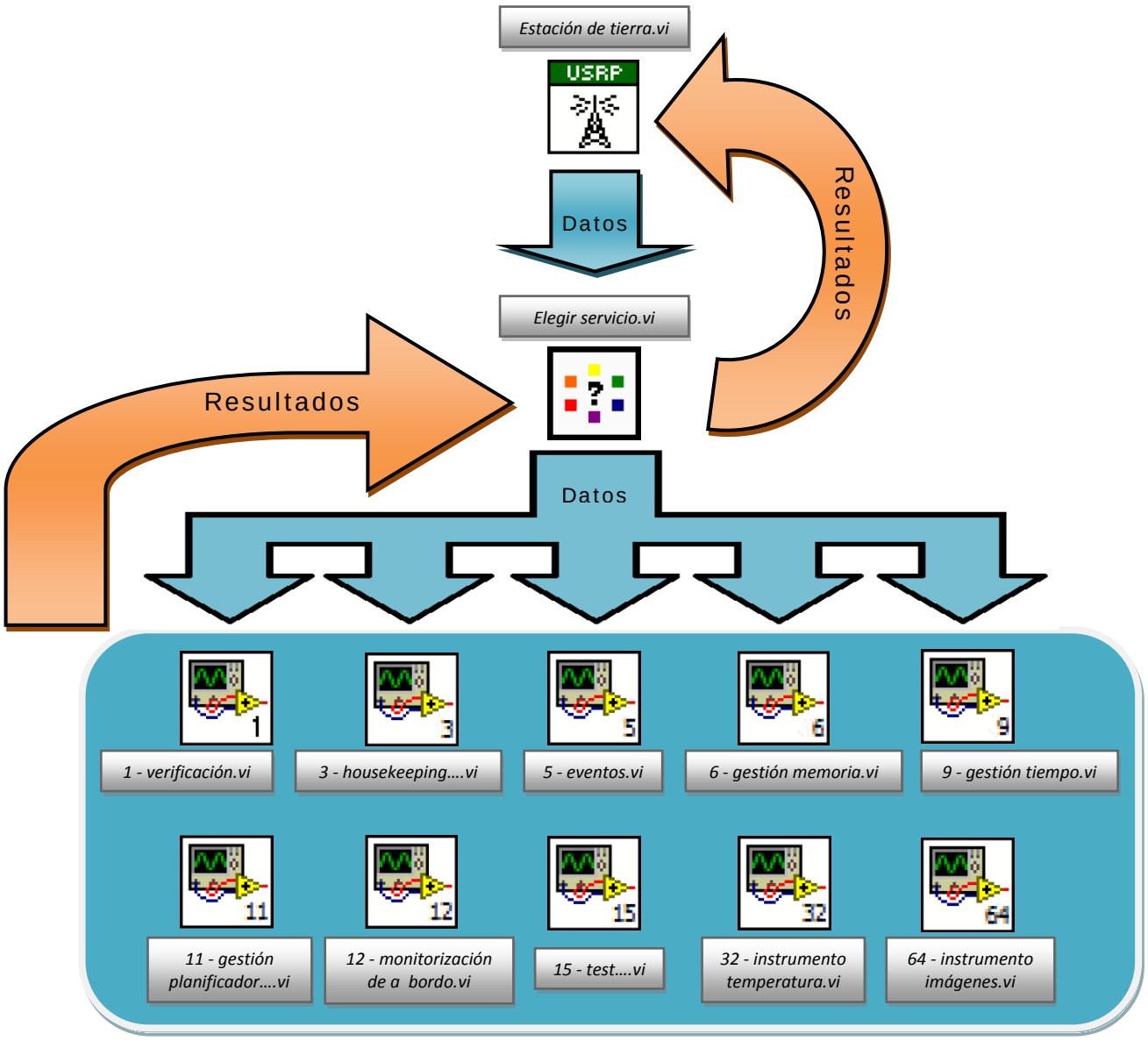


Figura 5.3: Esquema jerárquico de los módulos

Además de estos módulos, el programa utiliza otros módulos secundarios denominados SubVI's. Son denominados así ya que no realizan ninguno de los servicios "directamente", sino que son módulos auxiliares que son utilizados desde los principales. Éstos son:

- *Binario a decimal (SubVI).vi*: Este módulo se encarga de transformar los arrays de bits en binario a números en decimal. Es utilizado con gran frecuencia desde todos los módulos.

- *Escribir log (SubVI).vi*: Este módulo es el encargado de crear un fichero de texto y escribir en él la información que le llega. Es utilizado para llevar a cabo un diario –en inglés: log– en el que se guarda todo lo ocurrido en la estación de tierra, así como la fecha y hora. Este módulo también es utilizado con mucha frecuencia en todos los demás módulos.

Si se desea profundizar un poco más, estos dos módulos auxiliares vienen explicados de forma detallada en el Anexo C de la memoria.

A continuación se analiza de forma más detallada cada uno de los módulos principales mencionados anteriormente describiendo cómo se ha llevado a cabo la programación de cada uno de ellos mediante la plataforma LabVIEW [14] [15] [16].

5.2.1. Estación de tierra.vi

Este módulo es el módulo principal del programa ya que, como se puede observar en la Figura 5.3, todos los demás módulos están subordinados a él. Es el que se encarga tanto de la recepción de paquetes como de mostrar los resultados obtenidos. Además, el panel frontal de este módulo es con el que el usuario va a tratar directamente y donde va a poder ver los resultados obtenidos.

Antes de comenzar se ha de aclarar que este módulo no ha sido desarrollado completamente por los miembros del proyecto, sino que se ha utilizado como base un módulo disponible en los foros [38] de LabVIEW que permite enviar y recibir, y se ha modificado para adaptarlo al presente proyecto. Esto se hizo así porque, aunque el programa LabVIEW [14] [15] [16] cuenta con una serie de librerías que permiten la transmisión y recepción, no se consiguió que funcionaran ya que no se lograba que los dos transceptores NI USRP [7] – [10] se sincronizaran. Por este motivo se buscó en los foros algún otro módulo que hiciera esto y se encontró el que finalmente se ha utilizado, que consigue que los transceptores se sincronicen añadiendo al comienzo de cada paquete una secuencia de sincronización.

Si se observa el panel frontal de este módulo se puede ver que está formado por varias pestañas. Cada una de estas pestañas aporta información sobre un tema en concreto y son fácilmente interpretables por el usuario. Las pestañas de las que se dispone son las siguientes:

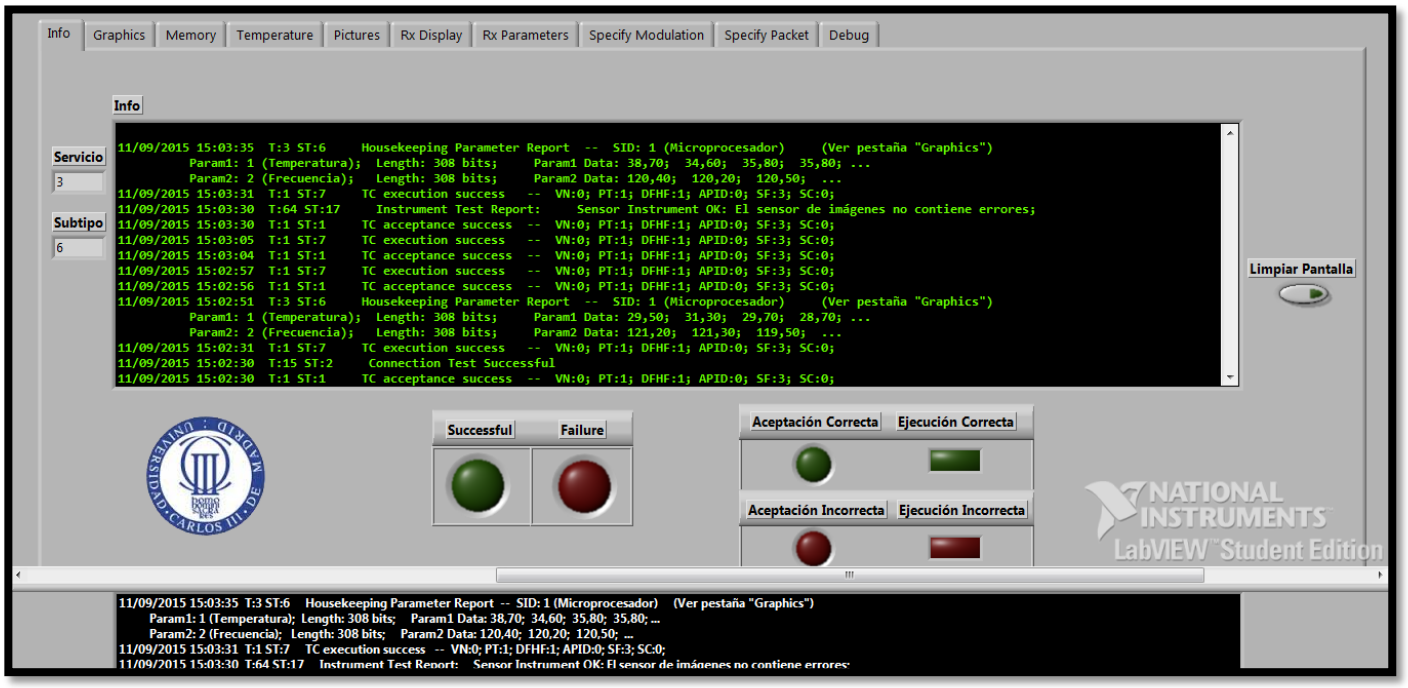


Figura 5.4: Pestaña de información

Esta primera pestaña llamada *Info* es una de las más importantes, ya que es en la que se muestran todos los resultados del programa. Es decir, esta pestaña es donde el usuario puede ver todo lo que el transceptor que simula el satélite está enviando.

Cuenta con una pantalla de texto donde aparece toda la información de todo lo que está ocurriendo en el satélite, y además cuenta con varios indicadores luminosos. Los dos de la izquierda se encenderán en función de si la acción realizada por el satélite se ha realizado con éxito (Successful) o fracaso (Failure). Cuando se encienda la luz roja de *Failure* también sonará un sonido repetido cuatro veces para avisar al usuario del error. Los otros cuatro indicadores luminosos de la derecha se corresponden con el servicio 1 de verificación y se encenderán cuando el telecomando enviado desde la estación de tierra haya sido correctamente o erróneamente aceptado y cuando haya sido correctamente o erróneamente ejecutado. Esta pantalla también indica el tipo y subtipo de servicio al que pertenece el paquete recibido.

Además cuenta con un botón que permite limpiar la pantalla y dejarla otra vez en negro, aunque toda la información mostrada y que ha sido guardada en el registro (*log*) no se pierde.

Figura 5.5: Pestaña de gráficas

En esta pestaña (Graphics) se recogen los datos del servicio 3 de housekeeping. En las gráficas se muestran los valores que se han recopilado y que el satélite ha enviado a la estación de tierra. Cada una de las cuatro gráficas está dedicada a un parámetro concreto. Esta pestaña funciona únicamente para este servicio, ya que para los demás servicios las gráficas no se utilizan y permanecen inactivas.

Figura 5.6: Pestaña de memoria

Esta pestaña (Memory) sirve exclusivamente para el servicio 6, encargado de la gestión de la memoria. En la pestaña se muestra una porción de la memoria que el satélite envía a la estación de tierra en el comando de telemetría (6,3) llamado "Memory dump", del cual se hablará más adelante. Esta pestaña es simplemente una muestra para que el usuario vea cómo es la memoria del satélite y para comprobar que la memoria recibida es igual a la enviada.

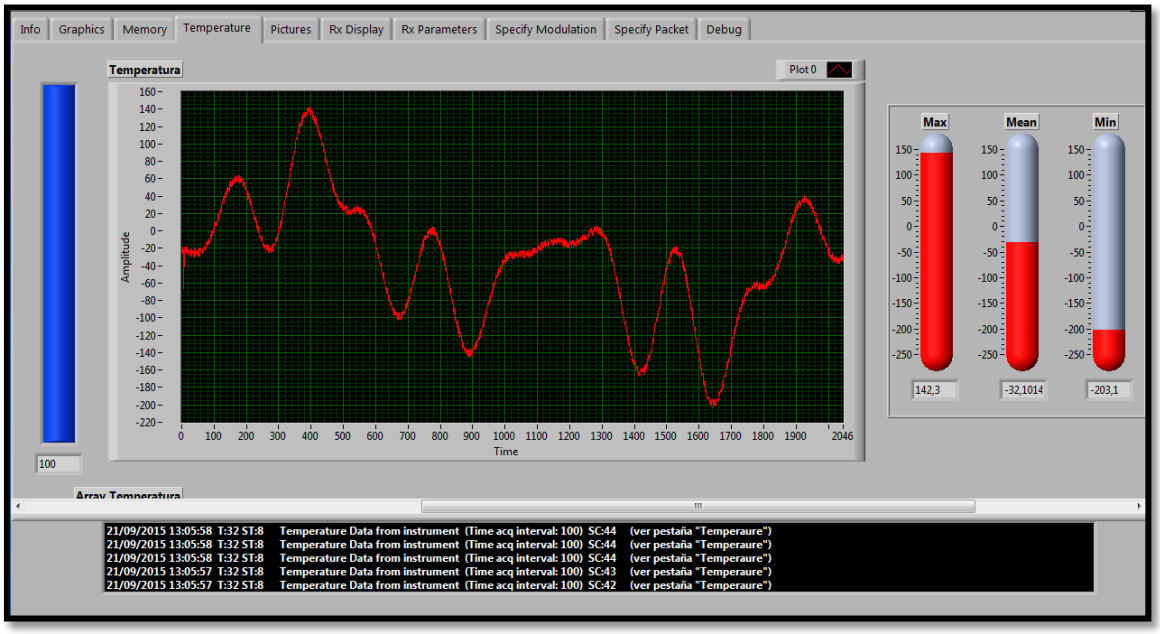


Figura 5.7: Pestaña de temperatura

En esta pestaña (Temperature) se muestra toda la información relativa al servicio 32, el encargado de gestionar la temperatura. En la pestaña se muestra un gráfico con todas las medidas de temperatura que han sido enviadas desde el satélite, y tres termómetros que indican las temperaturas máxima y mínima registradas por el satélite y la media de los datos enviados.

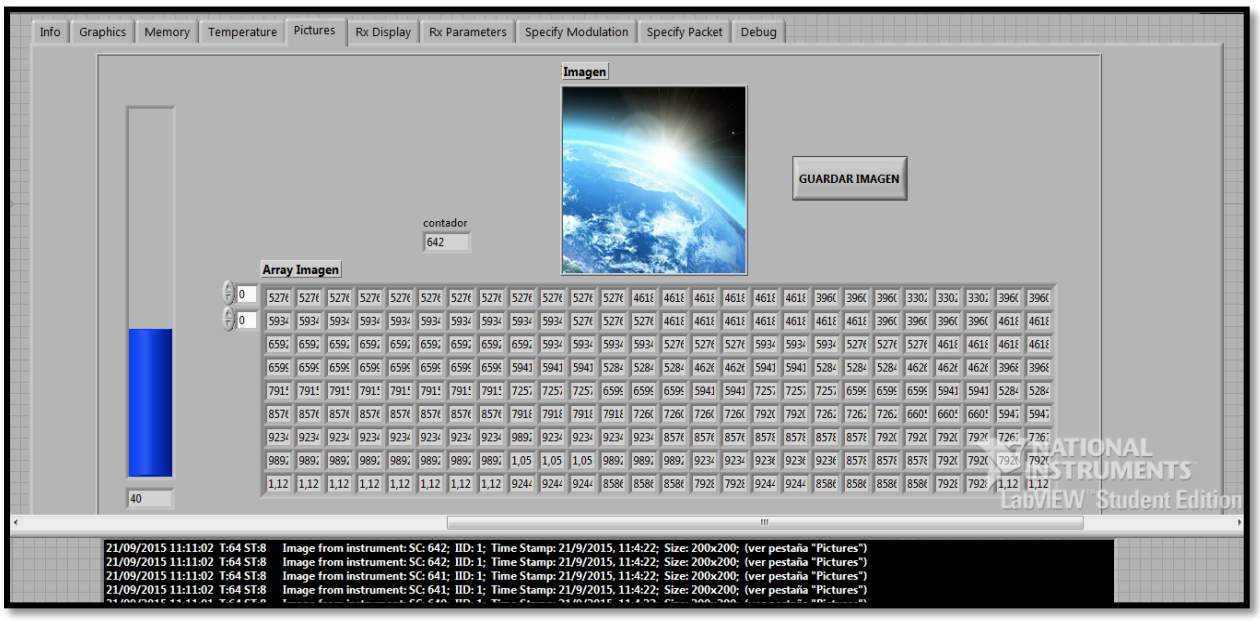


Figura 5.8: Pestaña de imágenes

En esta pestaña (Pictures) el usuario puede observar los resultados del servicio 64 del proyecto, el relativo a las imágenes. En esta pantalla se puede ver la imagen recibida que ha sido enviada desde el transceptor NI USRP [7] – [10] que simula el satélite, el contador que muestra el número de paquetes recibidos de la imagen, y el array bidimensional con los valores en RGB de la imagen. Como se comentará más adelante en el apartado 4.2.12, las imágenes que se van a utilizar en este proyecto son de tamaño 200x200 píxeles, ya que una imagen de tamaño superior tarda un tiempo excesivo en transmitirse. Además, esta pestaña cuenta también con un botón para guardar la imagen en el ordenador.

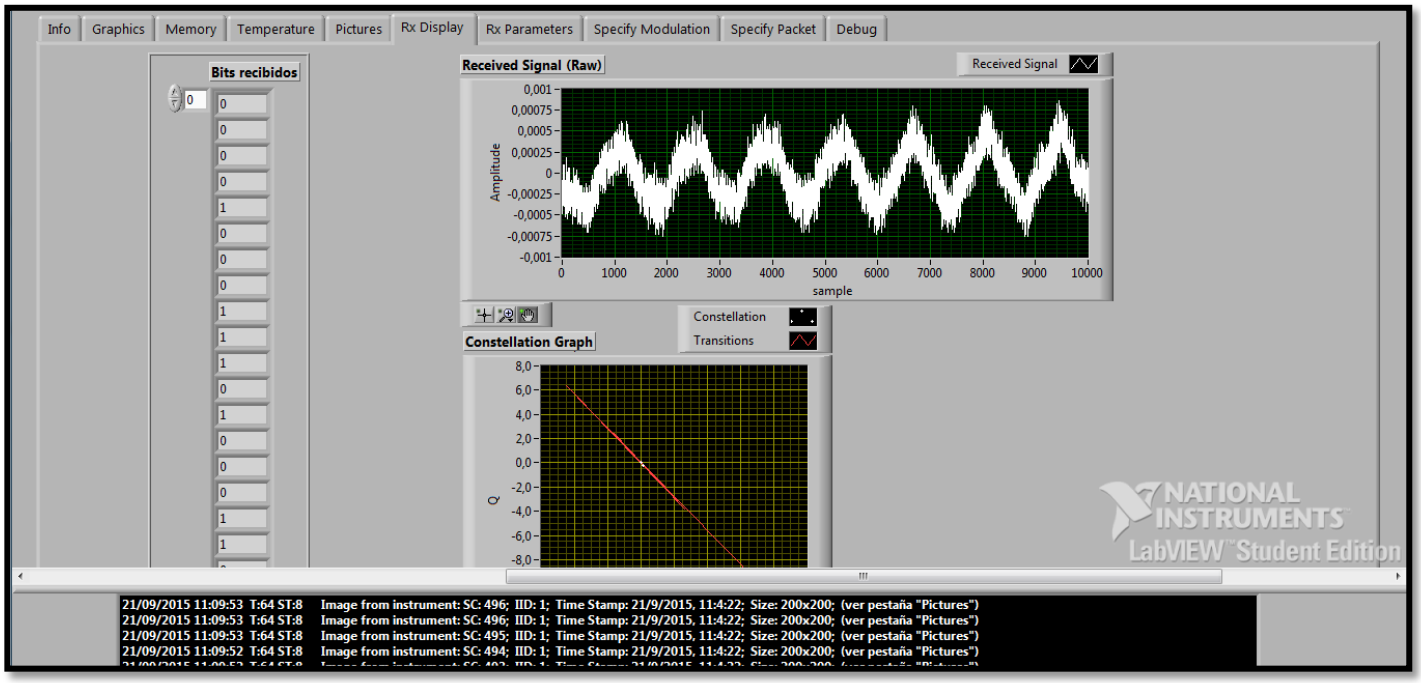


Figura 5.9: Pestaña de la pantalla de recepción

En esta pestaña (Rx Display) se puede observar la señal recibida de tres formas distintas: en primer lugar se puede ver la señal pura (sin modificar) que llega a la antena del transceptor; en segundo lugar se puede observar la señal transformada a forma de constelación; y en tercer lugar podemos ver la señal transformada ya a bits de información.

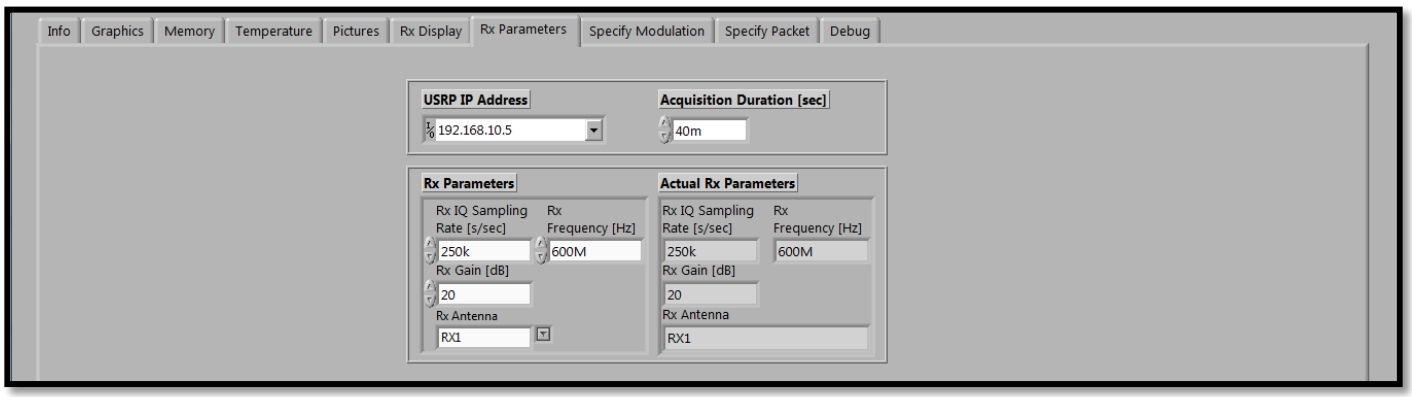


Figura 5.10: Pestaña de parámetros de recepción

En esta pestaña (Rx Parameters) se configuran los parámetros de recepción del transceptor NI URSP [7] – [10]. En primer lugar, se han de introducir la dirección IP del transceptor y el tiempo de adquisición. Después se han de configurar la tasa de muestreo, la frecuencia de funcionamiento del transceptor, la ganancia de recepción y la antena por la que debe recibir el transceptor. Se ha elegido la frecuencia de funcionamiento a 600 MHz ya que esa frecuencia no está ocupada en el Cuadro Nacional del Frecuencias [11] [12]. Se recomienda que los valores de los parámetros de recepción sean iguales que los que aparecen en la Figura 5.10 para un correcto funcionamiento del programa.

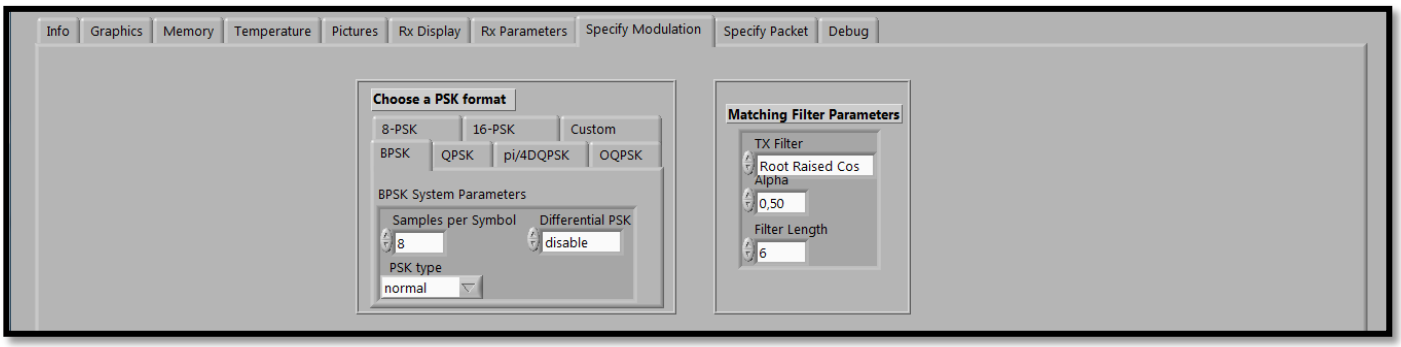


Figura 5.11: Pestaña de especificación de la modulación

La función de la siguiente pestaña (Specify Modulation) es la de poder configurar tanto la modulación que se desea utilizar, como los parámetros del filtro. Aunque se puede elegir el tipo de modulación que se quiera, se recomienda utilizar la modulación BPSK ya que es la más simple de todas y, en las pruebas que se han realizado, la que menos errores presenta. En cuanto a los parámetros del filtro, se recomienda asignar los valores que se observan en la Figura 5.11.

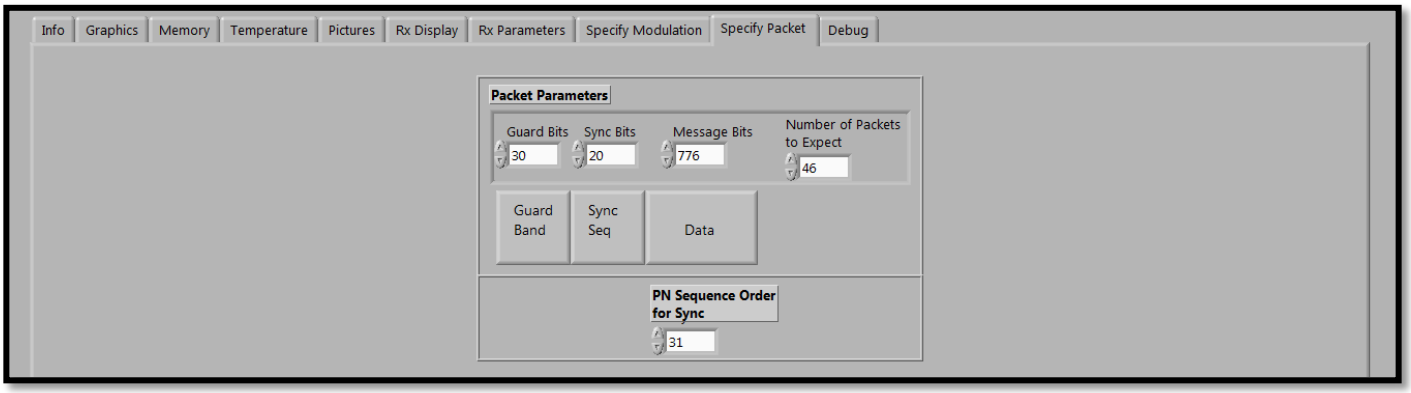


Figura 5.12: Pestaña de especificación de paquete

Esta pestaña (Specify Packet) sirve para configurar todo lo relativo a los paquetes. Se puede configurar el número de bits de guarda que se colocarán delante del paquete, el número de bits de sincronización que se utilizarán para sincronizar los transceptores, el número de bits por paquete, y el orden de la secuencia PN para la sincronización. El número de bits por paquete se ha

determinado de ese modo para hacer más fácil la transmisión y recepción de imágenes (esto se explicará más adelante en el apartado 5.2.12), por lo que ese número no se debe modificar. El número de bits de guarda y de sincronización se han determinado mediante pruebas, y esos son con los que mejores resultados se obtienen.

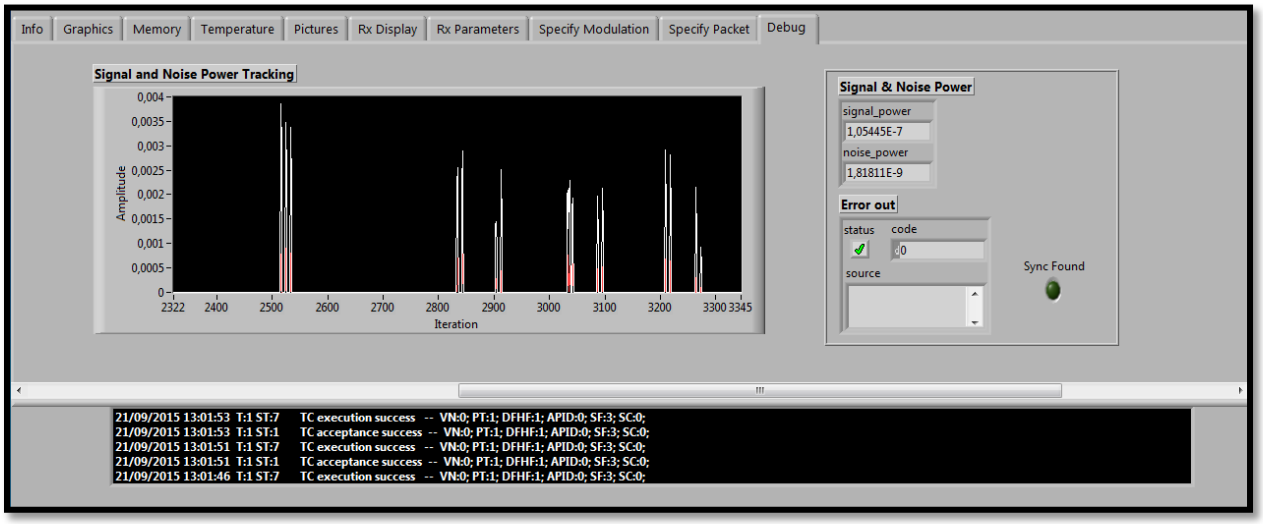


Figura 5.13: Pestaña de debug

Esta pestaña (Debug) permite ver la relación señal a ruido en cada momento de la recepción. Sirve fundamentalmente para ver cuándo le está llegando al transceptor señal con información y, por lo tanto, cuántos paquetes está recibiendo el transceptor. De este modo, se puede comprobar si al transceptor le está llegando señal desde el otro transceptor/satélite.

Además de estas pestañas, en la parte de abajo de la interfaz se muestra un cuadro de texto común a todas las pestañas. Este cuadro de texto muestra las cinco primeras líneas del cuadro de texto principal *Info*. La finalidad de esto es que el usuario pueda ver la información que se está recibiendo independientemente de la pestaña en la que se encuentre, para evitar tener que estar cambiando constantemente de pestaña.

Ahora en la siguiente parte se pasa a analizar el código de este módulo para comprender cómo funciona. Pero, antes de comenzar, simplemente aclarar que como este módulo no ha sido completamente programado por nosotros, no entraremos mucho en detalle de cómo está hecho ya que la memoria se ciñe únicamente al trabajo realizado por los alumnos. No obstante, sí que se comentará brevemente el funcionamiento del módulo para comprender cómo funciona.

Para empezar, en la página siguiente se puede observar en la Figura 5.14 el código en LabVIEW [14] [15] [16] del programa.

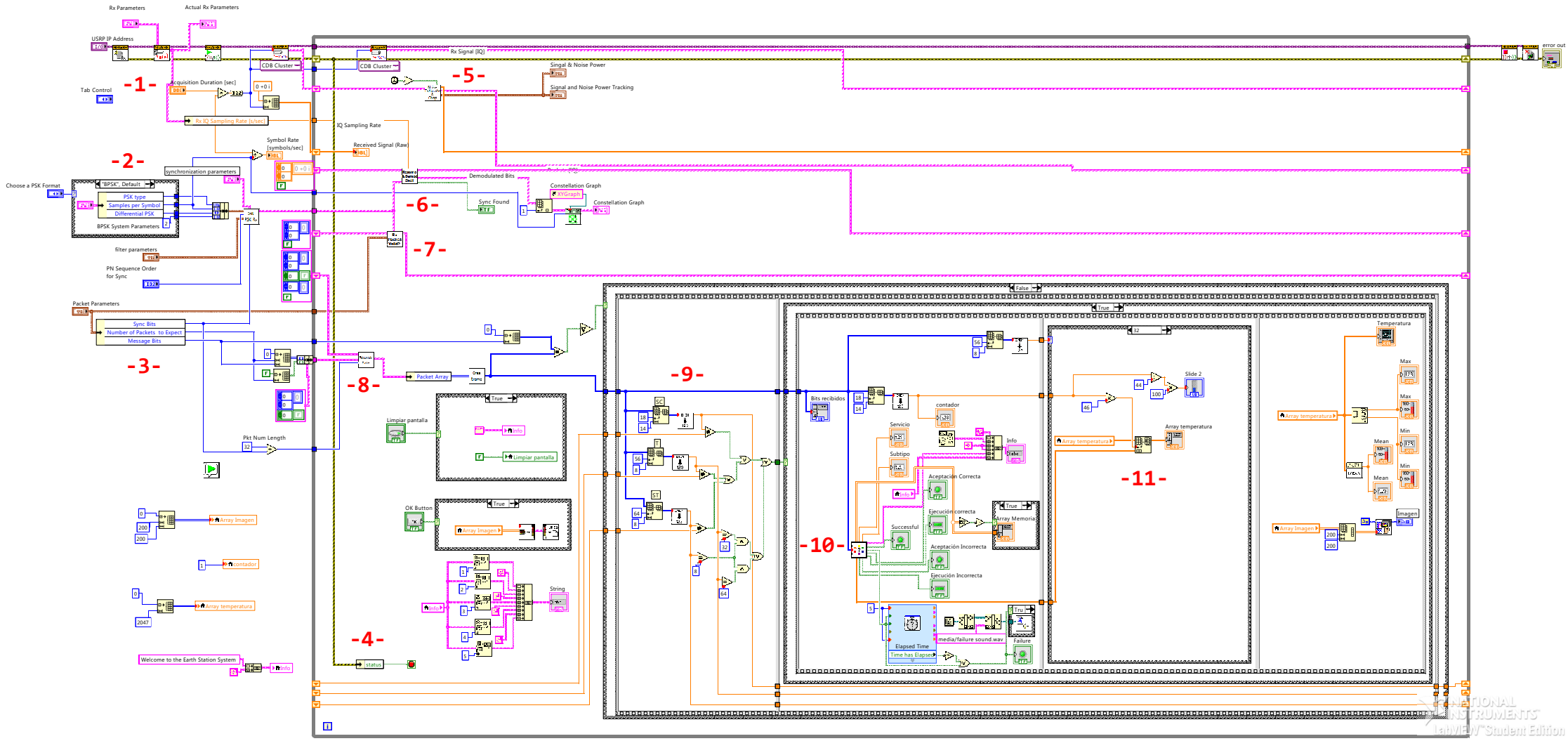


Figura 5.14: ***Estación de tierra.vi***

Como se puede ver en la Figura 5.14, este módulo tiene como entradas todos los parámetros de configuración mencionados anteriormente, y como salidas los indicadores de texto, luminosos, de imágenes y gráficas que también se comentaron anteriormente.

En primer lugar, el programa toma la dirección IP indicada en el panel frontal y los parámetros de recepción (*Rx Parameters*) y configura el transceptor NI USRP [7] – [10]. (Véase Nº1 en la Figura 5.14). Una vez configurados, el programa empieza a recibir los datos del transceptor.

En segundo lugar, el programa toma los parámetros de configuración de la modulación elegidos en la pestaña *Specify Modulation* así como el tipo de modulación elegida, para que después pueda demodular la señal. (Véase Nº2 en la Figura 5.14).

En tercer lugar, el programa también toma los parámetros indicados en la pestaña *Specify Packet*, donde se elige el número de bits por paquete, de guarda y de sincronización, para que después pueda recuperar el paquete original. (Véase Nº3 en la Figura 5.14).

Una vez hecho esto, el programa entra en el bucle principal. Este bucle se está ejecutando continuamente y su única condición de parada es que se produzca un error en el transceptor NI USRP [7] – [10]. (Véase Nº4 en la Figura 5.14).

Una vez dentro del bucle, los datos que llegan desde el transceptor son analizados y se calcula la relación señal a ruido de la señal entrante. (Véase Nº5 en la Figura 5.14). De esta forma el programa distingue entre la parte de la señal que es información de la que es ruido.

A continuación, ya con la parte de la señal que es información, se procede a la demodulación de la misma. (Véase Nº6 en la Figura 5.14). La demodulación se realiza con los parámetros que tomó el programa en el tercer paso antes mencionado, y forma de este modo la constelación de la señal.

Una vez demodulada la señal, se pasa a comprobar la secuencia de sincronización para ver si la señal es válida o no. (Véase Nº7 en la Figura 5.14). Tras comprobar que la señal es correcta, se procede a la reconstrucción de la trama original, obteniendo así el paquete enviado. (Véase Nº8 en la Figura 5.14).

Una vez obtenido el paquete original, se comprueba que los campos *Sequence Count*, *Packet Type* y *Packet Subtype* sean distintos de los del paquete recibido en anterior lugar (véase Nº9 en la Figura 5.14). Esto es así porque, como se comentará más adelante en el capítulo 6, cada paquete enviado por el satélite se transmite repetido cuatro veces para garantizar la llegada del paquete. Por lo tanto si los tres parámetros antes mencionados son iguales a los del paquete anterior quiere decir que el paquete recibido es una copia del anterior y por tanto no se ejecutará. Sin embargo, para los servicios de gestión de temperatura e

imágenes (32 y 64) esto no se utiliza ya que lo que interesa es que llegue la mayor cantidad de datos posible, aunque se repitan.

Finalmente, una vez comprobado que el paquete no es una repetición, éste se pasa al módulo *Elegir servicio.vi* (véase Nº10 en la Figura 5.14) que se encargará de interpretarlo y de determinar a qué tipo y subtipo de servicio pertenece. Este módulo también devuelve los resultados tras ejecutar el servicio correspondiente al paquete y los envía por las salidas para informar al usuario. Estas salidas son: un cuadro de texto (*Info)* en el que se escribe toda la información de lo que está sucediendo en cada momento; seis indicadores luminosos que informan sobre el éxito o fracaso de las diferentes acciones llevadas a cabo; cinco gráficas que muestran los datos de housekeeping y los datos de temperatura; un array bidimensional que muestra una porción de la memoria; y una imagen que muestra las fotografías enviadas por el satélite.

Además de esto, hay otra parte del código que es utilizada solo por los servicios 32 y 64 de temperatura e imágenes respectivamente. Esto es debido a que en estos servicios es necesario enviar la información en más de un paquete, por lo que hay que juntar varios paquetes antes de interpretar la información. (Véase Nº11 en la Figura 5.14). En el caso del servicio 64, la imagen es enviada en varios paquetes y ha de juntarse otra vez para poder ser representada; y en caso del servicio 32, los datos de temperatura son enviados también en varios paquetes y han de juntarse otra vez para poder representarlos en la gráfica. Esta parte del programa será descrita de forma más detallada en los apartados dedicados a los módulos de los respectivos servicios.

5.2.2. Elegir servicio.vi

Este módulo es el encargado de leer las cabeceras del paquete y de determinar a qué tipo de servicio pertenece. Como entrada recibe desde el módulo superior (Estación de tierra.vi) el paquete reconstruido, y como salida transmite los resultados que le llegan desde los módulos inferiores que implementan las funciones de los servicios, como se puede ver en la Figura 5.3.

Este módulo es también uno de los más importantes del programa, ya que se encarga tanto de comprobar que el paquete recibido es correcto, como de llamar al módulo correspondiente en función del servicio al que pertenece el paquete. El símbolo de este módulo, y que lo identifica, es el siguiente:

Figura 5.15: Símbolo de *Elegir servicio.vi*

En la siguiente página se puede ver en la Figura 5.16 el diagrama de bloques del módulo, y después se explica detalladamente el funcionamiento del mismo.

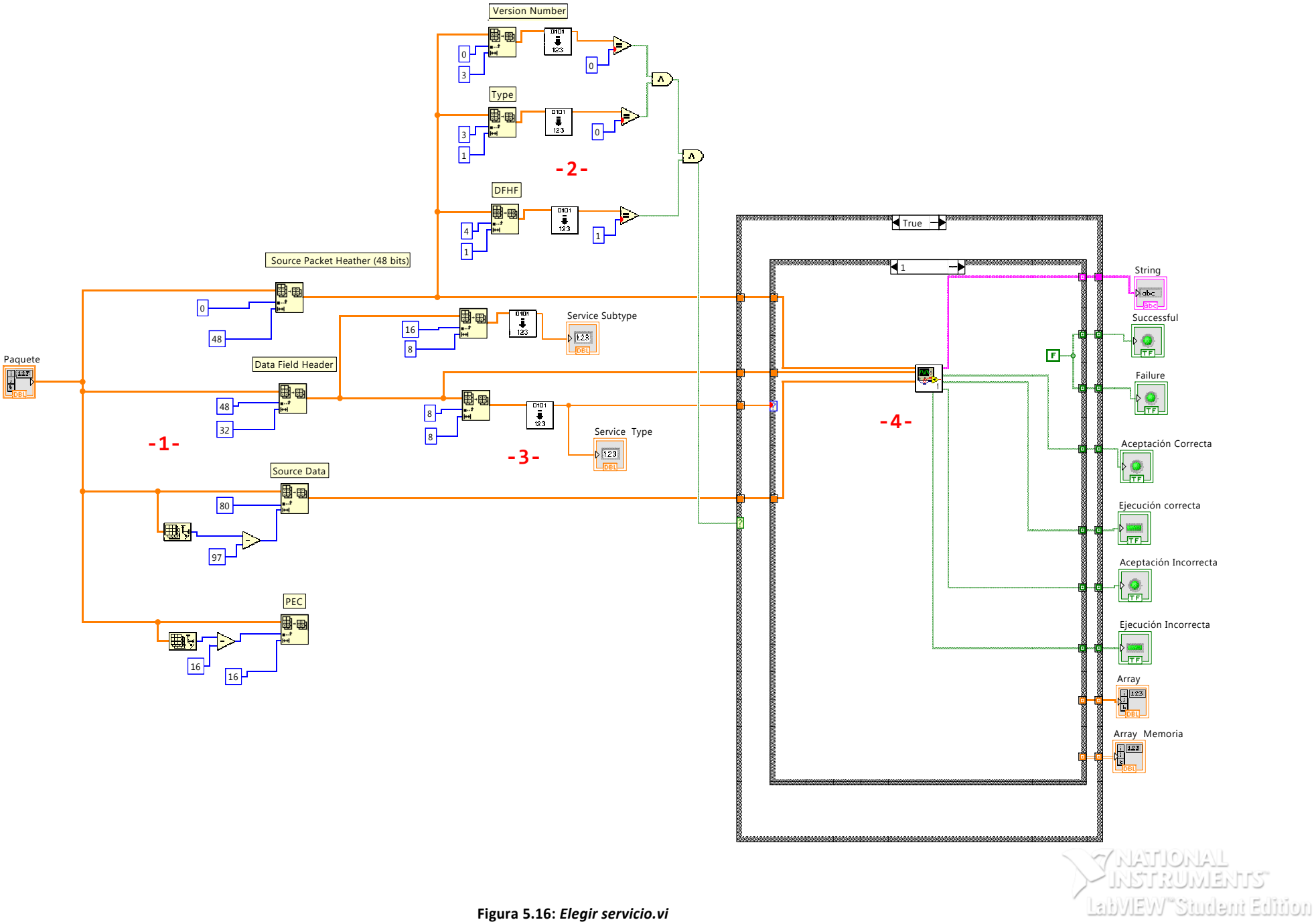


Figura 5.16: ***Elegir servicio.vi***

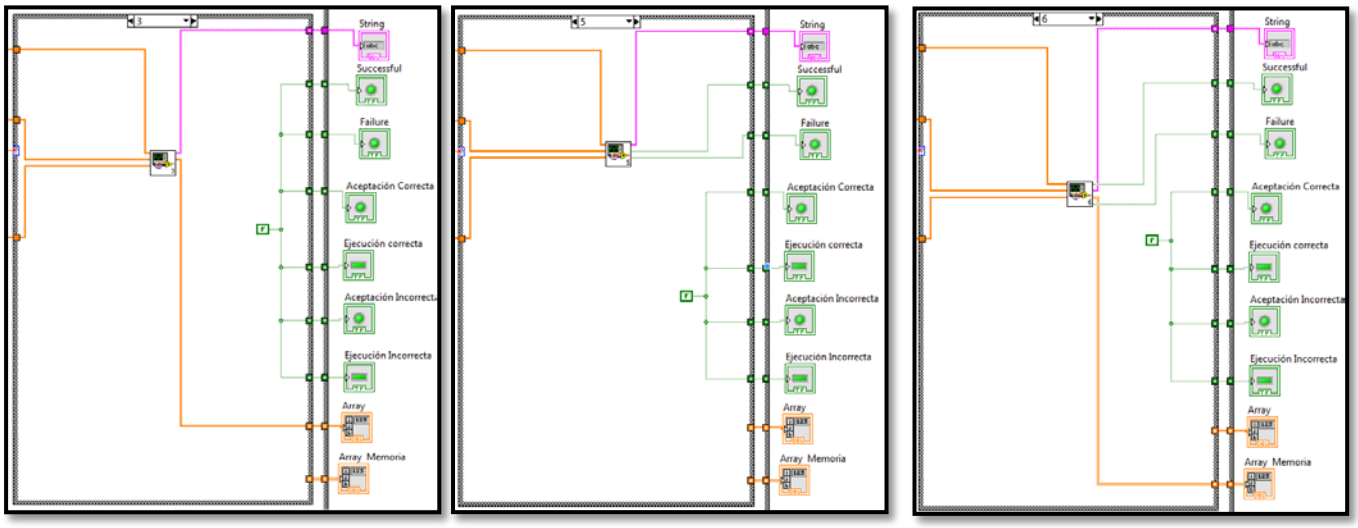
Como se puede observar en la Figura 5.16, este módulo lo primero que hace es tomar el paquete que le llega como entrada y separar las cabeceras, dividiendo el paquete en cuatro partes: *Packet Header* (cabecera del paquete), *Data Field Header* (cabecera del campo de datos), *Source Data* (datos) y *Packet Error Control* (control de errores de paquetes). (Véase Nº1 en la Figura 5.16).

Después de esto, lo siguiente que hace es comprobar que el paquete es correcto. Para ello lee la cabecera del paquete (*Packet Header*) y determina el valor del número de versión (*Version Number*), el tipo de paquete (*Type*) y el indicador de cabecera del campo de datos (*Data Field Header Flag*). (Véase Nº2 en la Figura 5.16). Estos tres valores deberán tener el valor '0', '0' y '1' respectivamente (véase apartado 5.1.1). Si los valores no son los correctos, el programa emitirá un mensaje de "paquete erróneo", mientras que si el paquete es correcto continuará la ejecución normal del programa.

Una vez comprobado que el paquete es correcto, lo siguiente que hace el programa es determinar a qué tipo de servicio pertenece. Para ello lee el campo *Service Type* (tipo de servicio) de la cabecera *Data Field Header*. (Véase Nº3 en la Figura 5.16).

Una vez conocido el tipo de servicio del paquete, el programa entra en una estructura tipo *Case* que se encarga de llamar al módulo correspondiente en función del tipo de servicio al que pertenece el paquete. Y después conecta las salidas de los módulos con los indicadores que corresponden a cada servicio. (Véase Nº4 en la Figura 5.16).

La configuración de las salidas en función del tipo de servicio puede verse a continuación en la Figura 5.17.



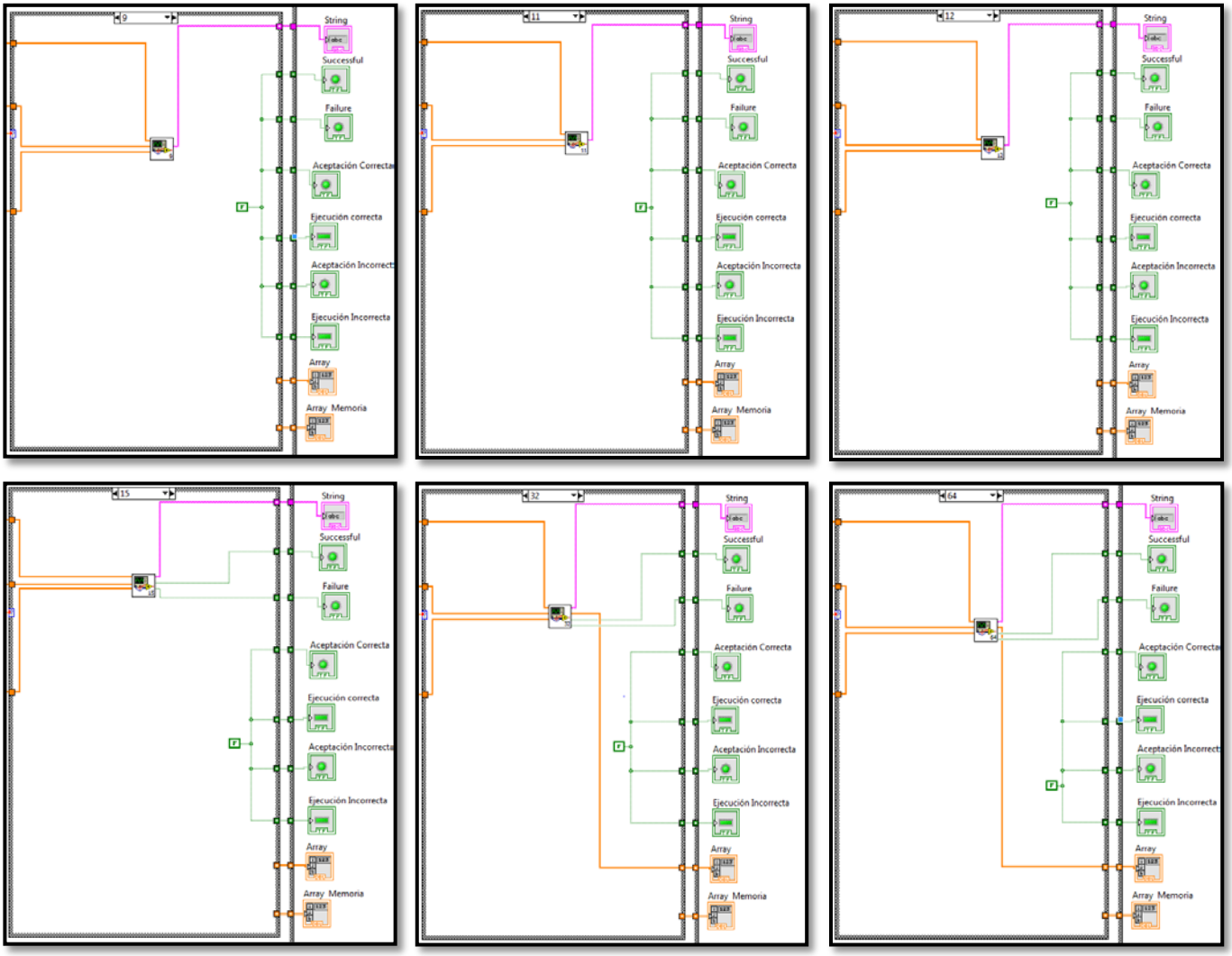


Figura 5.17: Casos de la estructura *Case*

El caso en el que el tipo de servicio es 1 no se ha incluido en la Figura 5.17 porque puede verse en la Figura 5.16.

Como se puede observar en la Figura 5.17, no todos los servicios tienen las mismas salidas y, por lo tanto, no todos los servicios utilizan los mismos indicadores. Esto depende de cada servicio y se explica detalladamente en el apartado correspondiente a cada módulo. Sin embargo, las entradas a todos los módulos sí son siempre las mismas. Estas son: en primer lugar la cabecera del paquete (*Packet Header*), en segundo lugar la cabecera del campo de datos (*Data Field Header*) y en tercer lugar los datos (*Source Data*).

Finalmente, en la imagen general del módulo (Figura 5.16) puede observarse también que se separan del resto del paquete los 16 últimos bits. Éstos se corresponden con el campo de control de errores (*Packet Error Control*). Como se mencionó en el apartado 5.1.1, este campo se deja como futura mejora, por lo que de momento el programa no realiza ninguna función con esos bits; simplemente los separa teniendo en cuenta su presencia.

5.2.3. 1 - verificación.vi

Este módulo es el encargado de implementar las funcionalidades del servicio 1, cuyo cometido es la verificación de los telecomandos que el satélite ha recibido. El símbolo del módulo es el siguiente:

Figura 5.18: Símbolo de *1 - verificación.vi*

Recibe como entradas las tres partes del paquete antes mencionadas (*Packet Header, Data Field Header* y *Source Data*) y tiene como salidas cuatro indicadores luminosos que informan acerca de si la aceptación y ejecución del telecomando han sido correctas o no, y una salida de texto por donde se informa del usuario de los detalles de lo que ocurre.

Este servicio tiene cuatro subtipos, y los cuatro son funciones de telemetría, por lo que se han implementado todos. Éstos se explican a continuación, pero antes, obsérvese la Figura 5.19 de la siguiente página donde se puede ver el código (diagrama de bloques) del módulo.

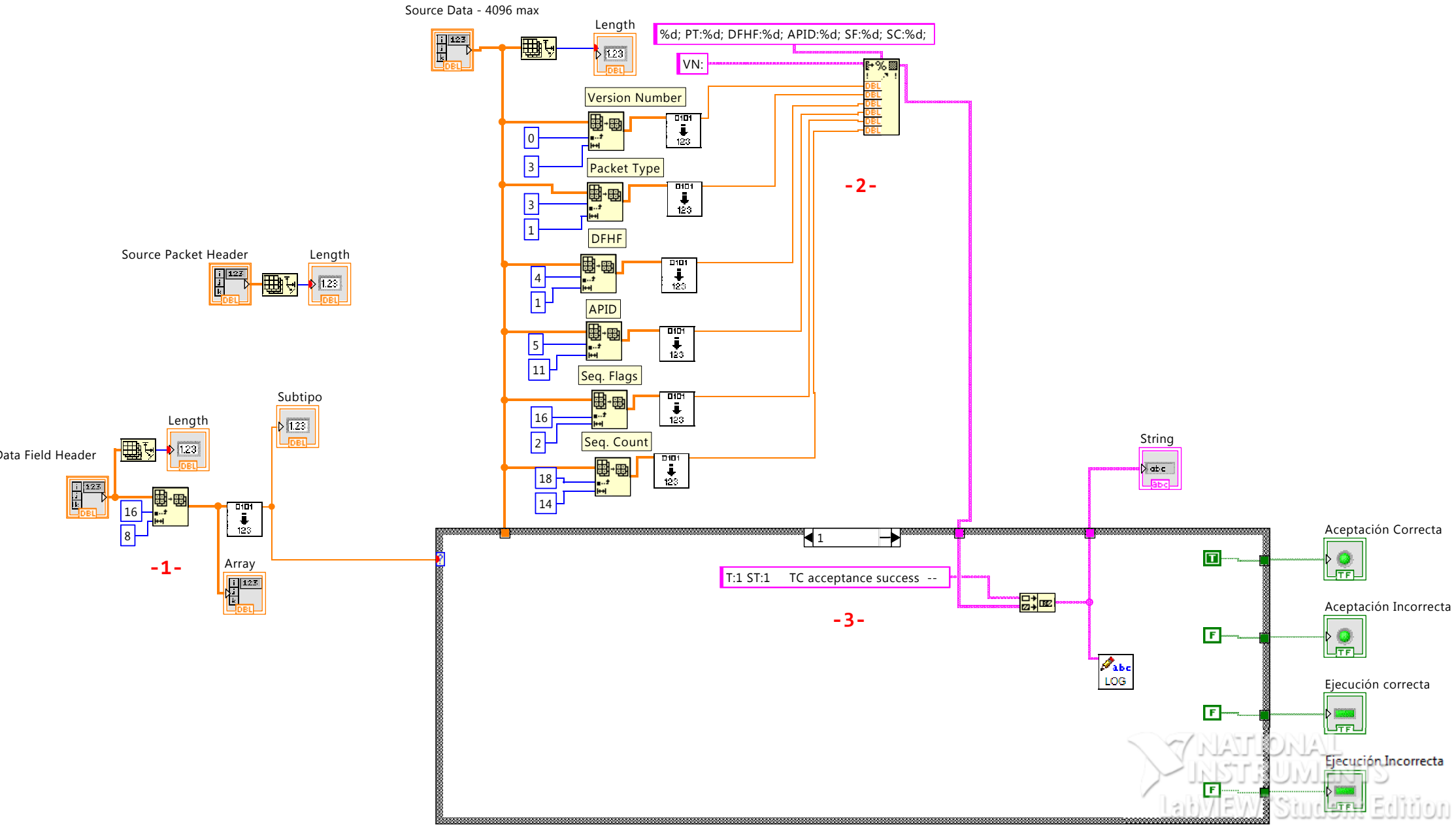


Figura 5.19: ***1 - verificación.vi***

Como se puede observar en la Figura 5.19, lo primero que hace el programa es leer el campo *Service Subtype* de la cabecera del campo de datos (*Data Field Header*) para poder ver a qué subtipo de servicio pertenece el paquete. (Véase Nº1 en la Figura 5.19). No se comprueba el tipo de servicio porque es obvio que si se ejecuta este módulo es porque previamente en el módulo superior (*Elegir servicio.vi*) se ha determinado que el paquete pertenece al servicio tipo 1.

Este módulo está implementado siguiendo las indicaciones del estándar de la ECSS [17]. Si se observa la explicación de las funciones de los cuatro subtipos del servicio, se puede ver que en los cuatro casos el satélite envía de vuelta a la estación de tierra en el campo de datos *Source Data* parte de la cabecera del paquete de telecomando que se le envió, concretamente los campos *Telecommand Packet ID* y *Packet Sequence Control*. (Si se desea ver la estructura del paquete de telecomando, en el Anexo A se encuentra un esquema del mismo).

Es por este motivo por el que, como se puede observar en la Figura 5.19, se toma el campo *Source Data*, se interpretan cada uno de los parámetros de la cabecera del telecomando que se ha enviado y se transforman a texto. (Véase Nº2 en la Figura 5.19).

Una vez determinado el subtipo de servicio al que pertenece el paquete e interpretada la información del campo de datos, el programa llega a una estructura de tipo *Case* que actuará de forma distinta en función del subtipo determinado previamente. Los subtipos de servicios implementados son los siguientes:

Telecommand acceptance success report (1,1)

Este comando de telemetría informa al usuario de que se ha aceptado correctamente el telecomando enviado previamente. La estructura del campo de datos (*Source Data*) es la misma que indica el estándar [17], y es la siguiente:

Telecommand Packet ID	Packet Sequence Control
2 octetos	2 octetos

Estos dos campos son una copia de los campos homónimos de la cabecera del paquete de telecomando que se envió previamente y son leídos por el programa antes de entrar en la estructura *Case*.

La implementación de este comando puede verse en la Figura 5.19 (véase Nº3). La implementación de este comando es sencilla ya que simplemente concatena al texto obtenido previamente del campo *Source Data* con "TC acceptance success" indicando que el telecomando se ha aceptado con éxito, y se escribe esta información en el *log* del programa. Además, el programa también enciende el indicador verde de "Aceptación correcta".

Telecommand acceptance failure report (1,2)

Este comando de telemetría es similar al anterior, solo que informa de que el telecomando enviado previamente no ha sido aceptado con éxito, es decir, que se ha producido un error en la aceptación. La estructura del campo de datos se ajusta también a la propuesta por el estándar [17], y es la siguiente:

Telecommand Packet ID	Packet Sequence Control	Code
2 octetos	2 octetos	1 octeto

Como se puede observar la estructura es la misma que en el caso anterior (se envía una copia de la cabecera del paquete de telecomando enviado), con la diferencia de que ahora se le añade un tercer campo *Code* en el que se indica el motivo del fallo. Los valores del campo *Code* y sus significados son los siguientes:

- 0 = APID ilegal.
- 1 = longitud de paquete incompleta o inválida.
- 2 = error de checksum.
- 3 = tipo de paquete ilegal.
- 4 = subtipo de paquete ilegal.
- 5 = datos de aplicación ilegales o inconsistentes.

La implementación de esto en LabVIEW [14] [15] [16] se ha hecho de la siguiente forma:

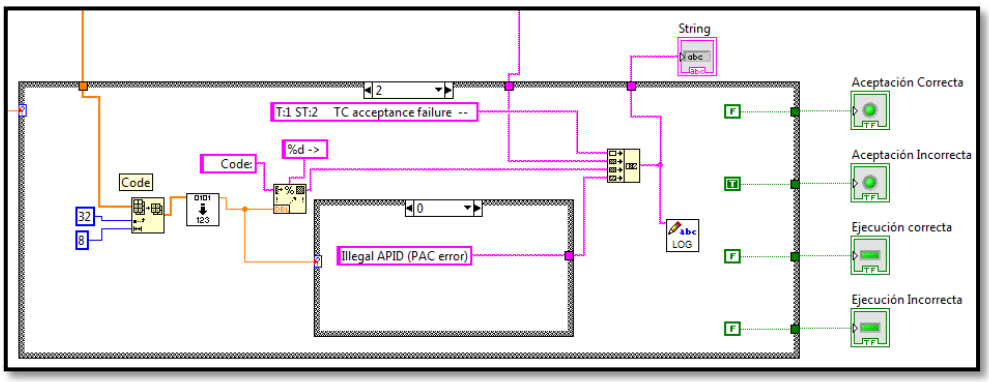


Figura 5.20: Implementación del servicio (1,2)

Como se puede observar, el programa lee el campo *Code* y en función del valor informa del error sucedido. Además también enciende el indicador luminoso correspondiente a "Aceptación Incorrecta".

Telecommand execution success report (1,7)

Este comando es exactamente igual que el (1,1) comentado anteriormente. La única diferencia es que ahora la recepción de este comando indica que la ejecución del telecomando enviado previamente ha sido correcta y se envía cuando la acción asociada al telecomando se ha finalizado con éxito.

La estructura del campo *Source Data* es también similar, siendo su estructura la siguiente:

Telecommand Packet ID	Packet Sequence Control
2 octetos	2 octetos

La implementación del comando en LabVIEW [14] [15] [16] también es igual, por lo que no es necesario adjuntar otra imagen del programa. Lo único en lo que difiere con el anterior es que ahora se mostrará por pantalla el mensaje de "TC execution success", y que se encenderá el indicador luminoso de "Ejecución correcta".

Telecommand execution failure report (1,8)

Este comando es también similar al anterior, solo que ahora informa de que ha habido un error en la ejecución del telecomando. El campo de datos es de igual modo una copia de la cabecera del paquete de telecomando, y su estructura es también la misma:

Telecommand Packet ID	Packet Sequence Control
2 octetos	2 octetos

La implementación también es idéntica, con la diferencia de que ahora se mostrará por pantalla el mensaje "TC execution failure" y se encenderá el indicador luminoso de "Ejecución Incorrecta".

5.2.4. 3 - housekeeping y diagnostico de datos.vi

Este módulo es el encargado de implementar las funcionalidades del servicio 3, que consiste en la gestión del housekeeping y del diagnóstico de datos. Consta de dos comandos de telemetría, los cuales se explicarán a continuación. También recibe como entradas los tres campos del paquete de telemetría, y tiene dos salidas: el panel de texto y un array con los datos de housekeeping para representarlos posteriormente de forma gráfica. El símbolo del módulo es:

Figura 5.21: Símbolo de *3 - housekeeping y diagnóstico de datos.vi*

A continuación se muestra en la figura 5.22 el código realizado en LabVIEW [14] [15] [16] que implementa el módulo.

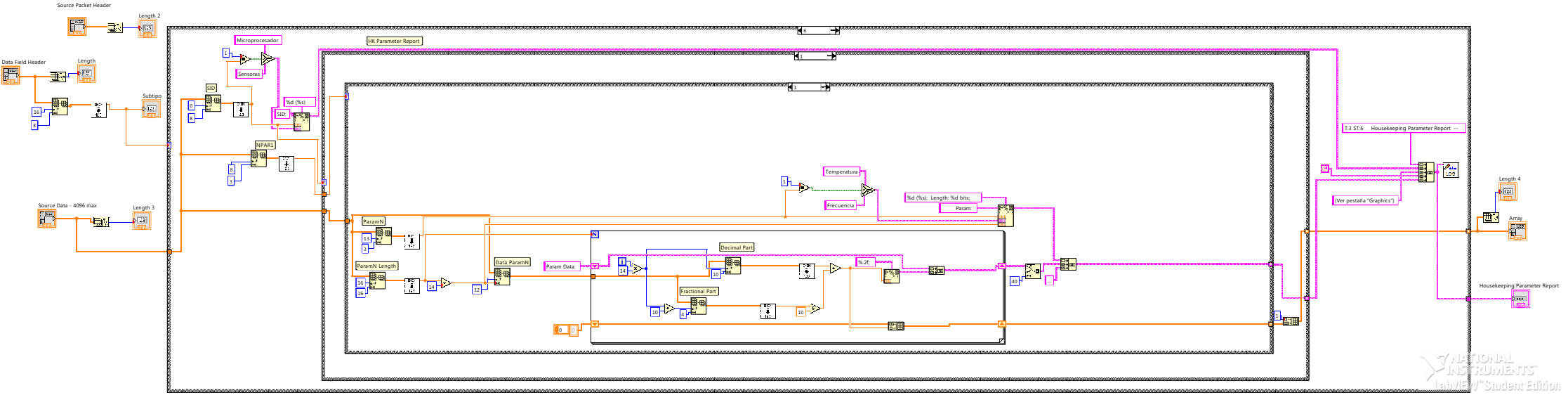


Figura 5.22: ***3 - housekeeping y diagnóstico de datos.vi***

Como se puede observar en la Figura 5.22, el programa lo primero que hace es leer el subtipo de servicio que se indica en la cabecera. Una vez hallado el subtipo al que pertenece, el programa llega a una estructura tipo *Case*, actuando de modo distinto en función del subtipo del paquete. Los comandos que se han implementado son los siguientes:

Housekeeping parameter report (3,6)

Este comando recibe la recolección de datos de housekeeping informando del estado de un determinado elemento del satélite. Este elemento viene identificado por un SID (*Structure Identification*). Además, cada elemento cuenta con varios parámetros a medir y se puede solicitar el envío de la información de uno de ellos o de varios.

Para este proyecto se ha simulado el comportamiento de dos de estos elementos de los que se recolectan datos de housekeeping. Estos son el microprocesador del satélite (identificado con el SID = 1), y los sensores de temperatura e imágenes (identificados con el SID = 9). Además, cada uno de estos elementos cuenta con dos parámetros de los que se obtienen datos: en el microprocesador (SID = 1) el Parámetro 1 hace referencia a la temperatura del microprocesador, y el Parámetro 2 a la frecuencia media de funcionamiento; en los sensores de temperatura e imágenes (SID = 9) el Parámetro 1 hace referencia al funcionamiento del sensor de temperatura, mientras que el Parámetro 2 lo hace al sensor de imágenes.

Cada paquete de telemetría enviado llevará información de uno de los dos SID's y de uno o de los dos parámetros del mismo. La estructura de este paquete de telemetría es la siguiente:

SID	NPAR1	Spare	ParamN	ParamN Length	Data ParamN
8 Bits	3 Bits	2 ó 7 Bits	3 Bits	16 Bits	Múltiplo de 8 Bits

Repetido NPAR1 veces

El campo *SID* indica el identificador del elemento sobre el que se están enviando los datos de housekeeping. El campo *NPAR1* indica el número de parámetros sobre los que se está enviando información, que puede ser 1 ó 2. El campo *Spare* son bits de relleno que tomarán el valor '0' y que serán dos bits si el número de parámetros (*NPAR1*) es uno, y siete bits si el número de parámetros (*NPAR1*) es dos. Esto se ha elegido de esta forma para que el paquete final tenga un número entero de octetos, es decir, que tenga un número de bits múltiplo de ocho.

Los siguientes tres campo se repetirán *NPAR1* veces en función del número de parámetros que haya en el paquete. El campo *ParamN* indica el número del parámetro del que se envía la información (puede ser 1 ó 2). *ParamN Length*

indica la longitud, en número de muestras, que tiene el campo *Data ParamN*. Finalmente, el campo *Data ParamN* contiene los datos del parámetro que se ha medido. Este último campo tendrá un formato distinto en función del elemento (*SID*) del que sea la información:

Si se trata del microprocesador (SID = 1), cada dato o muestra del mismo, bien sea de la temperatura o de la frecuencia, estará formado por 14 bits. Estos bits formarán un número fraccional indicando el valor de la temperatura o la frecuencia. Estos datos irán incluidos en el campo *Data ParamN* y tienen la siguiente estructura:

Decimal Part	Fractional Part
10 Bits	4 Bits

Si se trata de los sensores (SID = 9), cada dato estará formado por un único bit que indicará si el sensor en cuestión estaba activo o no en el momento en el que se midió. La estructura de este campo será por lo tanto:

State (‘0’/’1’)
1 Bit

Si observamos ahora la Figura 5.22 para ver como se ha implementado esto mediante el programa LabVIEW [14] [15] [16], vemos que hay dos estructuras *Case* anidadas. La primera de ellas actúa en función del identificador (SID) del elemento y la segunda procede en función del número de parámetros (NPAR1) que contiene el paquete. En la Figura 5.22 se muestra el caso en el que el SID es 1 y NPAR1 es 1.

Llegado a este punto, lo primero que hace el programa es leer el identificador del parámetro (*ParamN*) para saber de qué parámetro son los datos. Después lee el campo *ParamN Length* para saber el número de muestras que contiene el paquete, y calcula la longitud en bits multiplicando el número anterior por 14, ya que como se mencionó antes cada muestra o dato del SID 1 consta de 14 bits.

Una vez hecho esto, el programa entra en un bucle tipo *For* donde va leyendo los datos de 14 en 14 bits y los va transformando a números fraccionarios. Después de interpretar todos los datos, el programa los guarda en el *log* y los envía tanto en formato de texto como en el array.

Para el caso en el que se envían los datos de dos parámetros la implementación en similar a la anterior. Lo único que varía es que ahora se realiza el mismo procedimiento contado anteriormente, pero dos veces (una por cada parámetro).

Ahora para el caso de los sensores (SID = 9), la implementación es muy parecida a la anterior pero con la diferencia de que ahora cada muestra o dato

está formado únicamente por un bit que indica si el parámetro estaba encendido o no en el momento en el que se tomaron las muestras. Esto se ha implementado en LabVIEW [14] [15] [16] de la siguiente manera:

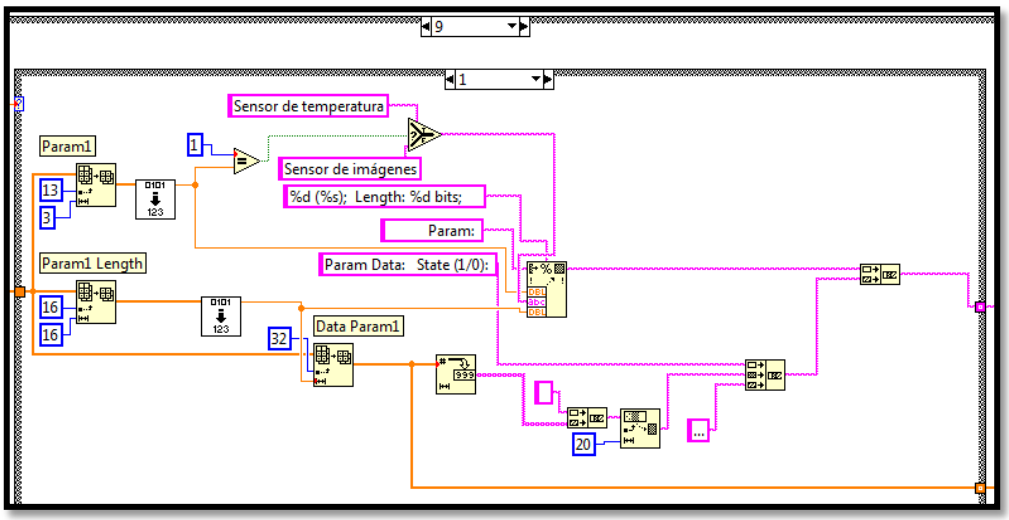


Figura 5.23: Implementación para el caso SID = 9 y NPAR1 = 1

Como se puede observar en la Figura 5.23, en este caso se procede de manera similar al anterior. Primero se leen el identificador del parámetro y la longitud de los datos y después se transforman los datos a texto para poder mostrarlos por pantalla posteriormente. También se envía el array de datos para después representarlos gráficamente.

Para el caso en el que hay datos de dos parámetros, la implementación es igual que en el caso anterior, con la diferencia de que se repite el mismo procedimiento dos veces. A continuación se muestra en la Figura 5.24 el código en LabVIEW [14] [15] [16] para este caso:

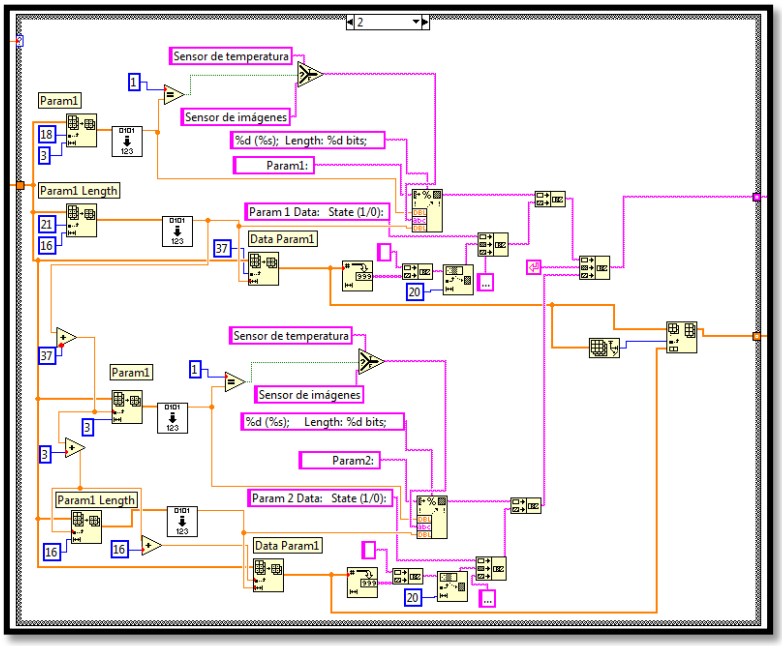


Figura 5.24: Implementación para el caso SID = 9 y NPAR1 = 2

Housekeeping report interval updated (3,9)

Con este comando simplemente se informa de que el intervalo con el que se envían los datos de housekeeping a la estación de tierra ha cambiado. Este comando de telemetría es enviado por el satélite como respuesta a los telecomandos (3,7) y (3,8) donde se da la orden al satélite de cambiar el período de recolección de datos. La estructura del campo de datos (*Source Data*) para este comando tiene la siguiente estructura:

SID	Period
8 Bits	24 Bits

El campo *SID* es el identificador del elemento en el que se ha modificado el período de recolección de datos. Y el campo *Period* corresponde al nuevo período (en segundos) de recolección de datos. Este campo consta de 24 bits y por lo tanto tiene un rango desde 0 a 16.777.215 (2^24-1) segundos, que equivalen a 194,18 días. La implementación de este comando en LabVIEW [14] [15] [16] se ha hecho de la siguiente forma:

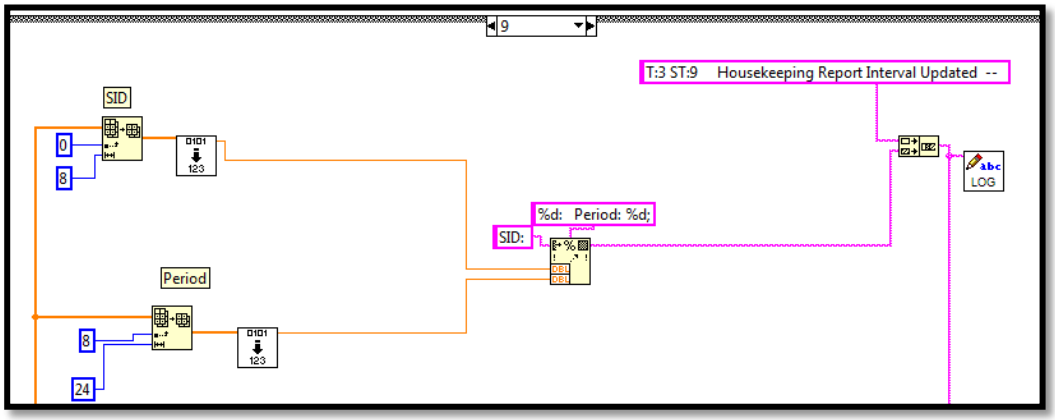


Figura 5.25: Implementación del servicio (3,9)

Como se puede observar en la Figura 5.25, el programa simplemente lee el SID y el nuevo período y lo envía por la salida de texto para que se muestre al usuario.

5.2.5. 5 - eventos.vi

Este módulo implementa las funcionalidades del servicio 5, encargado de gestionar los eventos ocurridos en el satélite. Este servicio cuenta con seis comandos de telemetría, los cuales se explican a continuación. El módulo cuenta con las tres mismas entradas que los demás: *Packet Header, Data Field Header* y *Source Data*; y tiene tres salidas: dos indicadores luminosos que muestran si ha habido éxito o fallo, y un panel de texto donde se informa al usuario de lo sucedido. La implementación de este módulo en LabVIEW [14] [15] [16] se puede ver en la Figura 5.26 de la página siguiente.

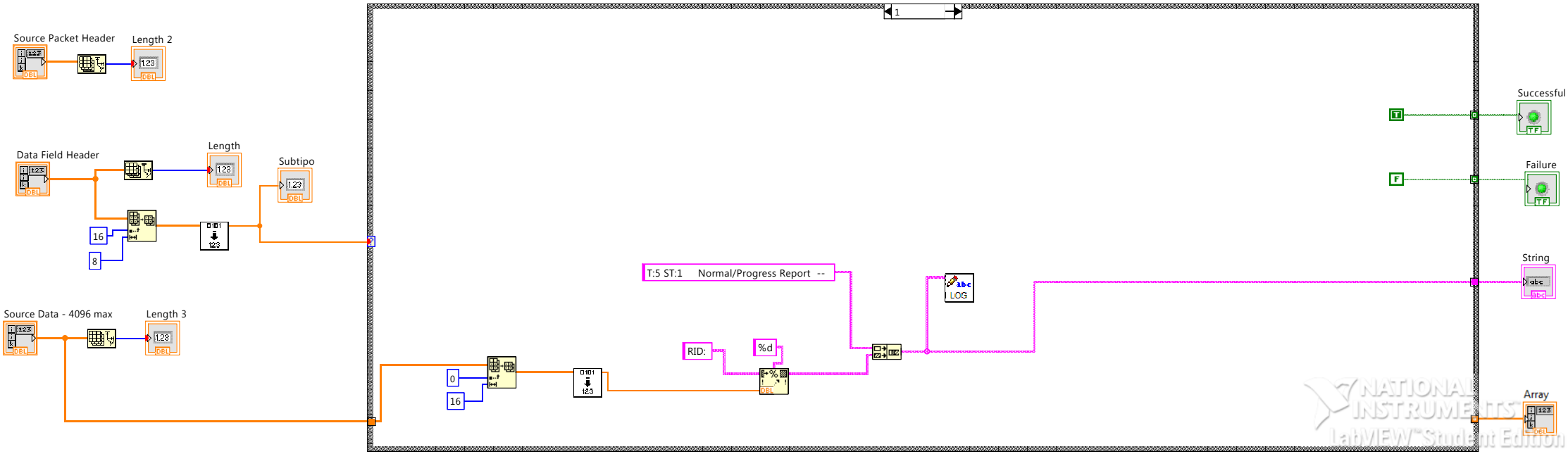


Figura 5.26: ***5 - eventos.vi***

Como se puede observar en la Figura 5.26, lo primero que hace el programa es leer el subtipo de servicio del paquete y en función de eso actúa de una determinada manera mediante una estructura *Case*.

Antes de proseguir, se ha de aclarar que en este servicio cada evento está asociado a un "identificador de reporte" –en inglés: Report ID (RID)– que identifica el evento y su severidad. Los eventos simulados en el satélite se detallan más adelante, cada uno en su correspondiente subtipo.

Normal progress report (5,1)

Este comando de telemetría se recibe cuando el satélite ha realizado una acción de forma autónoma que no ha producido ningún error. La estructura del campo de datos es la siguiente:

RID
16 Bits

Como se puede observar, en este caso solo se envía el identificador del evento (RID) que ha tenido un funcionamiento normal. La implementación de este comando se puede ver en la Figura 5.26 donde se puede observar que lo que hace el programa es leer este RID y enviarlo por la salida de texto (además de almacenarlo en el *log*). También se enciende el indicador luminoso de color verde indicando que no se ha producido ningún error y que tiene un funcionamiento normal.

Error / Anomaly report – Low severity – Warning (5,2)

Este comando de telemetría informa de que se ha producido un error en el satélite de baja severidad (warning). Los errores de baja severidad que se han simulado en el satélite y sus correspondientes identificadores son los siguientes:

- RID 7: Cambio de orientación del panel solar.
- RID 8: Bajo aumento de temperatura.
- RID 9: Borrado de sector de memoria RAM.

El formato del campo de datos es similar al del comando anterior:

RID
16 Bits

Al igual que en el caso anterior solo se envía el identificador del evento. La implementación de este comando en LabVIEW [14] [15] [16] se puede ver a continuación en la Figura 5.27.

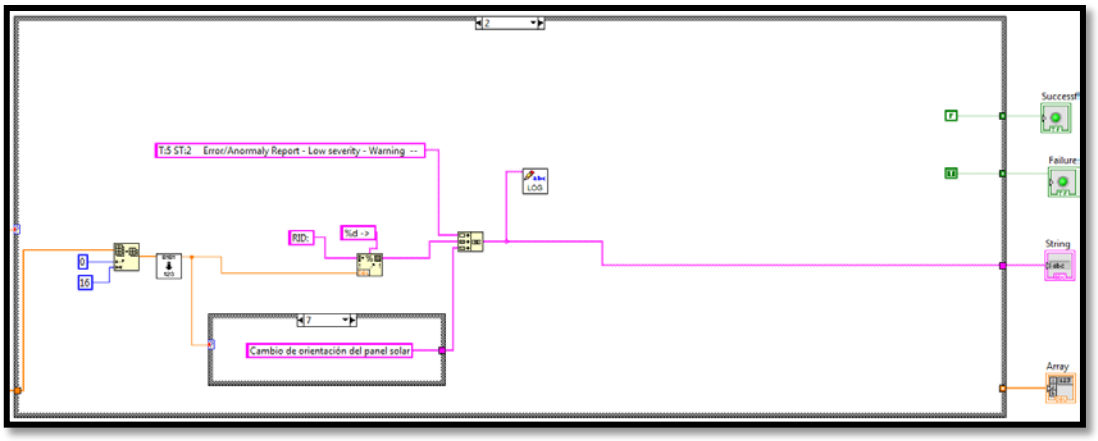


Figura 5.27: Implementación del servicio (5,2)

Como se puede observar en la Figura 5.27, el funcionamiento del programa es similar al del comando anterior, con la diferencia de que hay otra estructura *Case* anidada que identifica el evento ocurrido, y que ahora se enciende el indicador luminoso de color rojo (*Failure*) informando al usuario de que ha habido un error.

Error / Anomaly report – Medium severity – Ground action (5,3)

El siguiente comando de telemetría se recibe cuando se ha producido un error en el satélite de severidad media y que requerirá llevar a cabo una acción desde tierra. Los errores de severidad media que se han simulado en este proyecto y sus respectivos identificadores son los siguientes:

- RID 62: Fallo de un sector de memoria.
- RID 63: Temperatura alta en el instrumento de imágenes.
- RID 64: Repliegue del panel solar necesario.

La estructura del campo de datos es también similar a la del comando anterior, ya que solo está formada por el identificador del evento (RID).

RID
16 Bits

La implementación en LabVIEW [14] [15] [16] es idéntica a la del comando anterior, solo que ahora el mensaje de texto de salida informa de un error de severidad media.

Error / Anomaly report – High severity – On-board action (5,4)

Este comando de telemetría informa de que se ha producido un error en el satélite de alta gravedad, y que requerirá de una acción a bordo para solucionarlo. Los errores de alta severidad que se han simulado en este proyecto son:

- RID 85: Fallo del microprocesador.
- RID 86: Fallo en el sistema de alimentación.
- RID 87: Nivel de temperatura excesivo en el sensor.

Tanto la estructura del campo de datos como la implementación en LabVIEW [14] [15] [16] del programa son idénticas a las del comando anterior. La estructura es:

RID
16 Bits

Enabled event packets report (5,18)

En este comando de telemetría el satélite envía a la estación de tierra una lista de todos los eventos que tienen activa la generación de reportes, es decir, aquellos eventos cuyo RID está activo. La estructura del campo de datos para este comando es la siguiente.

NRID	RID (Repetido NRID veces)
16 Bits	16 Bits

El campo *NRID* indica el número de eventos (RID's) que se envían en la lista. El campo *RID* se repetirá *NRID* veces e indica los RID's que están activos en el satélite. La implementación de este comando en LabVIEW [14] [15] [16] se puede ver a continuación en la Figura 5.28.

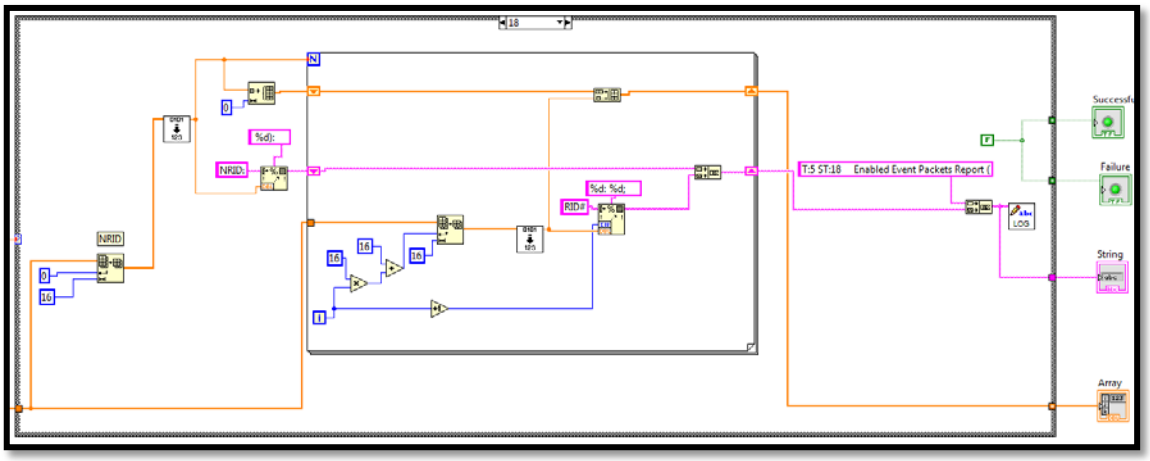


Figura 5.28: Implementación del servicio (5,18)

Como se puede observar en la Figura 5.28, lo que hace el programa en primer lugar es leer el campo *NRID* para saber el número de RID's que hay en la lista. Después el programa entra en un bucle tipo *For* en el que va leyendo los RID's de la lista de 16 en 16 bits. Por último el programa envía la lista por la salida de texto para informar al usuario.

Disabled event packets report (5,20)

Este comando es exactamente igual que el anterior, con la diferencia de que ahora el satélite envía una lista de los eventos que tienen desactivada la generación de reportes. Por este motivo tanto la estructura del campo de datos (*Source Data*) del paquete, como el código realizado en LabVIEW [14] [15] [16] para llevar a cabo la implementación son idénticos a los del comando anterior. Por

esa razón no es necesario adjuntar otra imagen del código del programa, ya que es idéntico al del comando anterior. La estructura del campo de datos donde se envía la lista de eventos inactivos también es similar. Ésta es:

NRID	RID (Repetido NRID veces)
16 Bits	16 Bits

Al igual que antes, el campo *NRID* indica el número de elementos de la lista, y el campo *RID* los identificadores de los eventos.

5.2.6. 6 - gestión memoria.vi

Este módulo implementa las funcionalidades del servicio 6, encargado de la gestión de la memoria. Este servicio cuenta con dos comandos de telemetría, los cuales se explicarán a continuación. El módulo cuenta con las mismas tres entradas que los demás (*Packet Header, Data Field Header* y *Source Data*) y consta de cuatro salidas: un panel de texto, un array bidimensional que muestra la memoria y dos indicadores luminosos. El símbolo del módulo es el siguiente:

Figura 5.29: Símbolo de *6 - gestión memoria.vi*

A continuación se muestra en la Figura 5.30 el código implementado mediante el programa LabVIEW [14] [15] [16]:

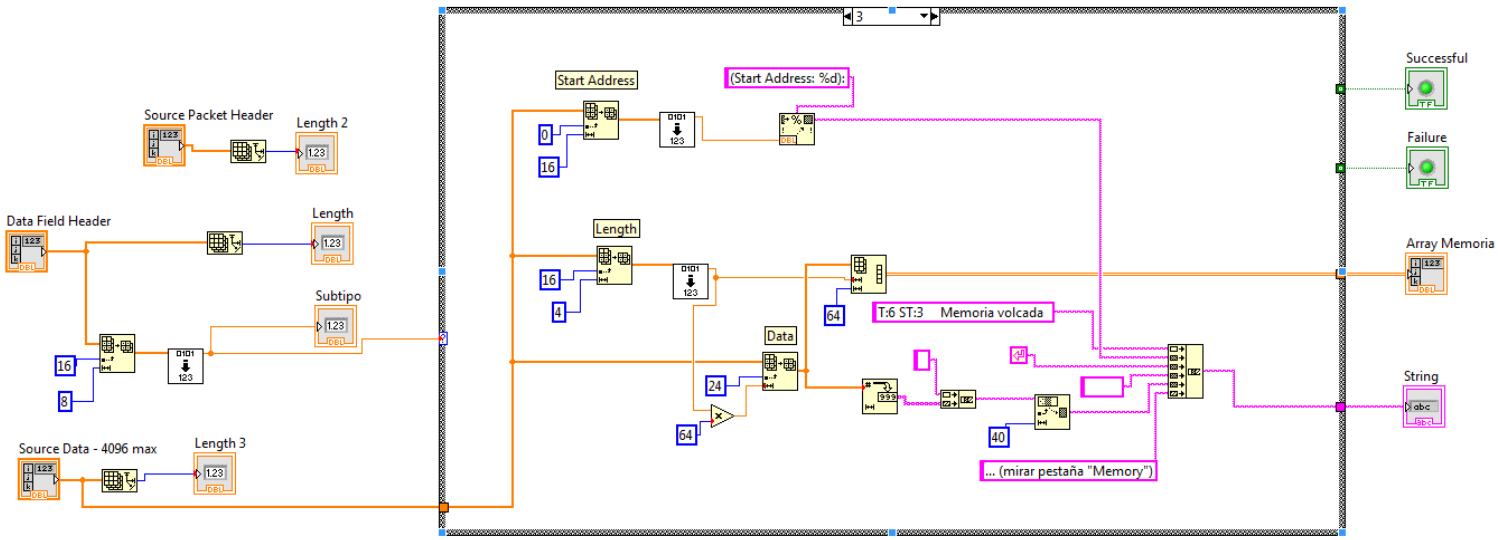


Figura 5.30: *6 - gestión memoria.vi*

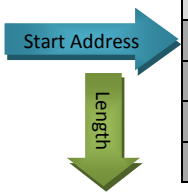
Como se puede observar en la Figura 5.30, el programa comienza leyendo el número de subtipo de servicio al que pertenece el paquete para actuar de una

manera determinada en función del mismo. Los comandos de telemetría implementados son los siguientes.

Memory dump (6,3)

Este comando lo envía el satélite como respuesta al telecomando "Dump memory area (6,2)". En este comando de telemetría se envía la porción de la memoria del satélite que se solicitó previamente desde la estación de tierra.

La memoria del satélite está formada por bloques de 64 bits por 2048. El satélite manda con este comando de telemetría una porción de esta memoria, indicando mediante el parámetro *Start Address* el comienzo del bloque y mediante el parámetro *Length* el número de filas del bloque.



	Bit 0	Bit 1	…	Bit 63
Bloque 1	X	X	X	X
Bloque 2	X	X	X	X
…	X	X	X	X
Bloque 2048	X	X	X	X

Figura 5.31: Esquema de un bloque de memoria

La estructura del campo de datos de este comando es la siguiente:

Start Address	Length	Spare	Data
16 Bits	4 Bits	4 Bits	Múltiplo de 8 Bits

El campo *Start Address* indica el comienzo de los datos. El campo *Length* indica el número de bloques que se envían. El campo *Spare* contiene cuatro bits con valor a '0' para conseguir un número entero de octetos. Y el campo *Data* contiene la información.

Como se puede ver en la Figura 5.31, lo que hace el programa es leer el campo *Data* donde se encuentra la memoria y lo almacena en un array que es enviado como salida. También transforma una parte de la memoria a texto para poder mostrarlo por pantalla al usuario.

Memory check report (6,5)

Este comando de telemetría es enviado por el satélite después de comprobar el estado de la memoria. Si el campo de datos contiene ocho bits con valor '0' indica que la memoria no contiene errores. Si por el contrario se envían con valor '1' quiere decir que hay un error en la memoria. La estructura del campo de datos (*Source Data*) para este comando de telemetría es la siguiente:

Memory OK
8 Bits '00000000'/'11111111'

La implementación de este comando en LabVIEW [14] [15] [16] es sencilla ya que simplemente se comprueba si el valor de los ocho bits del campo de datos son ceros o unos. Tras comprobar si la memoria contiene errores o no, el programa muestra un mensaje informando al usuario y enciende el indicador luminoso correspondiente. El código del programa (diagrama de bloques) se muestra a continuación en la Figura 5.32.

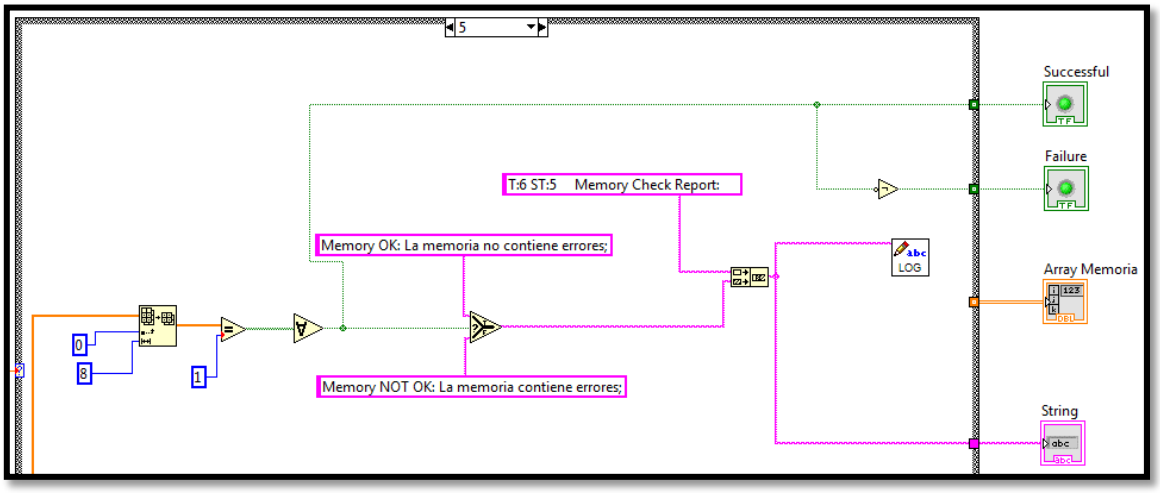


Figura 5.32: Implementación del servicio (6,5)

5.2.7. 9 - gestión del tiempo.vi

Este módulo implementa las funcionalidades del servicio 9, encargado de gestionar el tiempo. Cuenta solamente con un comando de telemetría. Tiene las mismas tres entradas que los demás módulos y solo tiene una salida, que es de texto. El símbolo de este módulo es:

Figura 5.33: Símbolo de *9 - gestión del tiempo.vi*

Al igual que en los demás módulos, lo primero que hace el programa es comprobar el subtipo de servicio para actuar en consecuencia. A continuación se muestra en la Figura 5.34 el código en LabVIEW [14] [15] [16] que implementa el programa.

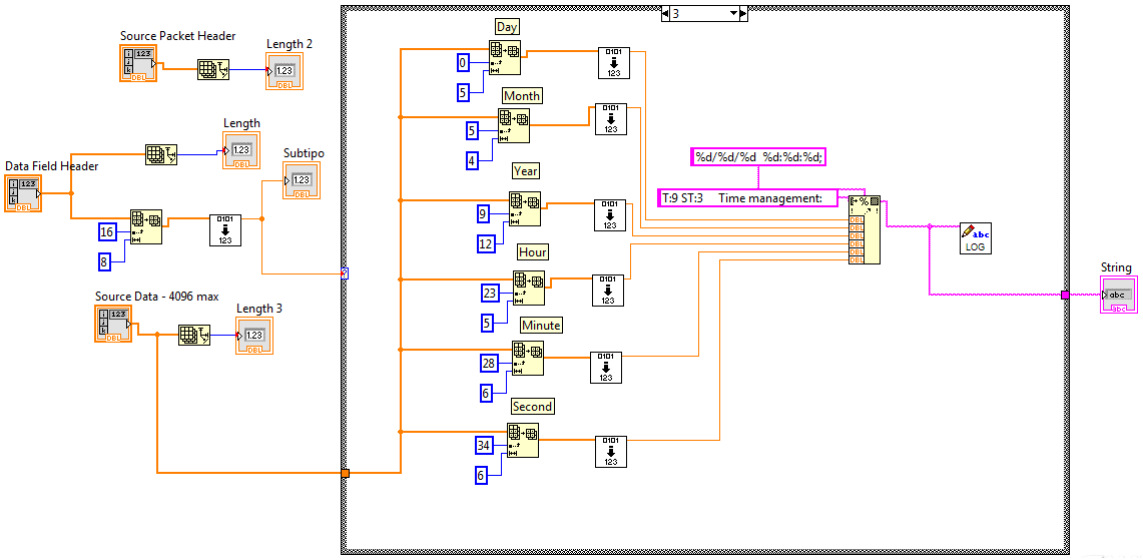


Figura 5.34: ***9 - gestión del tiempo.vi***

Time management (9,3)

En este comando de telemetría el satélite envía a la estación de tierra el tiempo (fecha y hora) que tiene el satélite en ese momento. La estructura del campo de datos para enviar esta información es la siguiente:

Day	Month	Year	Spare	Hour	Minute	Second
5 Bits	4 Bits	12 Bits	2 Bits	5 Bits	6 Bits	6 Bits

Cada uno de los parámetros que se indican en la tabla está codificado con el número de bits que se muestra debajo. En cuanto a la implementación del programa, que se puede ver en la Figura 5.34, simplemente se va leyendo cada uno de estos campos y se manda por la salida el resultado en forma de texto.

5.2.8. 11 - gestión del planificador de TC.vi

Este módulo es el encargado de llevar a cabo los comandos del servicio 11, cuyo cometido es la gestión del planificador de telecomandos. Este servicio cuenta solo con un comando de telemetría, encargado de mostrar una lista de los telecomandos planificados en el calendario de a bordo. El módulo cuenta con tres entradas (*Packet Header, Data Field Header* y *Source Data*), y con una única salida que es el panel de texto. Aunque este servicio solo cuenta con un comando de telemetría, el programa comprueba que éste sea el correcto. El símbolo de este módulo es el siguiente:

Figura 5.35: Símbolo de ***11 - gestión del planificador de TC.vi***

Al igual que en los demás módulos, lo primero que hace el programa es comprobar el subtipo de servicio al que pertenece el paquete de telemetría. Después de esta comprobación, el programa llega a una estructura tipo *Case* que funciona de forma distinta en función del subtipo: si el subtipo se corresponde con el 8, se ejecutará el programa que implementa la funcionalidad del servicio (11,8); si el subtipo es distinto al 8, el programa emitirá un mensaje de error informando al usuario de que el subtipo de servicio no existe.

A continuación se muestra en la Figura 5.36 el código en LabVIEW [14] [15] [16] del programa que lleva esto a cabo.

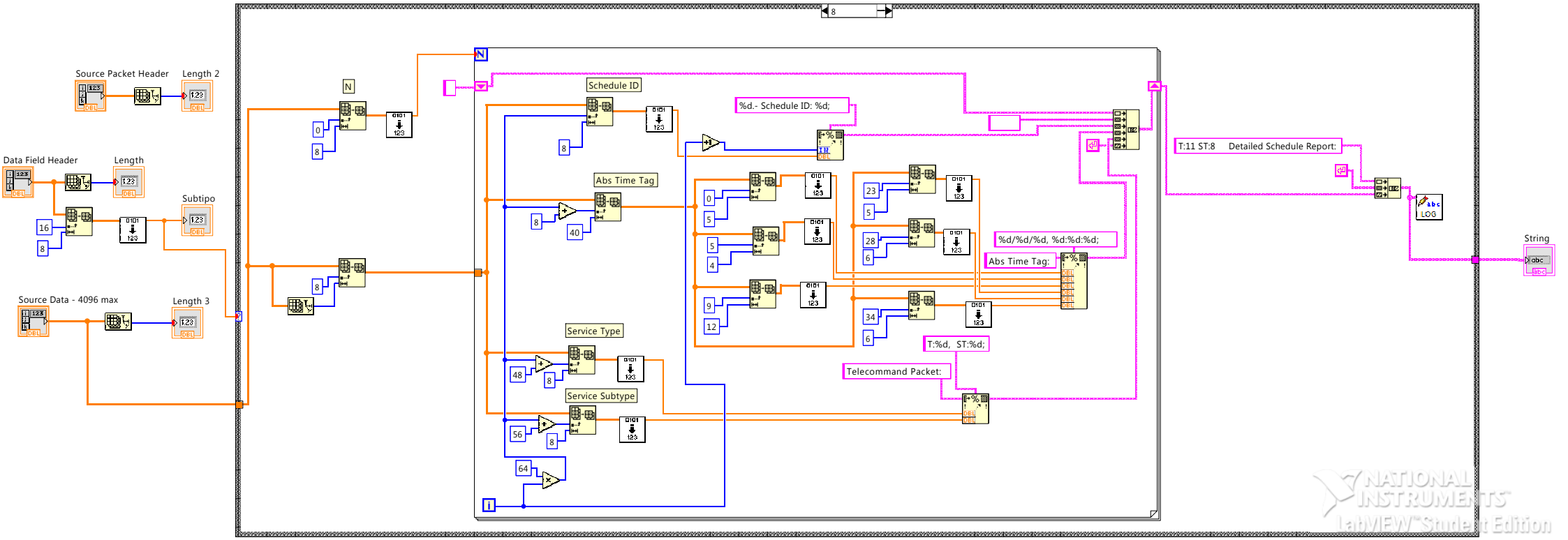


Figura 5.36: ***11 - gestión del planificador de TC.vi***

Detailed Schedule report (11,8)

En este comando de telemetría el satélite envía a la estación de tierra una lista de los telecomandos que tiene programados en el calendario para ejecutarse de forma automática. La estructura del campo de datos para este comando es:

N	Schedule ID	Abs Time Tag	TC Packet Length	Telecommand Packet
8 Bits	8 Bits	40 Bits	16 Bits	Múltiplo de 8 Bits

Repetido N veces

El campo *N* se corresponde con el número de telecomandos que hay en la lista. El campo *Schedule ID* contiene el identificador del telecomando. En el campo *Abs Time Tag* está incluido el tiempo en que está programado que se ejecute el telecomando. El campo *TC Packet Length* indica la longitud del siguiente campo que contiene el telecomando. Finalmente, el campo *Telecommand Packet* contiene el telecomando programado en el satélite. (En el Anexo A se encuentra un esquema de la estructura de un telecomando).

Como se puede observar en la Figura 5.36, lo que el programa hace en primer lugar es leer el campo *N* para ver el número de telecomandos que hay en la lista. Después el programa entra en un bucle tipo *For* que se repite *N* veces y va leyendo los distintos campos del comando. Después los transforma a texto, forma la lista y finalmente la envía por la salida de texto para que el usuario la vea.

5.2.9. 12 - monitorización a bordo.vi

Este módulo implementa las funcionalidades del servicio número 12. Éste es el encargado de monitorizar los parámetros de a bordo conforme a unas comprobaciones definidas desde la estación de tierra. Este servicio cuenta solamente con un comando de telemetría, encargado de imprimir la lista de parámetros monitorizados. Consta de las mismas tres entradas que los otros módulos y tiene una única salida de texto. El símbolo del módulo es:

Figura 5.37: Símbolo de *12 - monitorización a bordo.vi*

El programa lo primero que hace, al igual que en los demás módulos, es comprobar que el subtipo del paquete es el correcto mediante una estructura tipo Case. Si el subtipo es 10, se ejecutará el módulo; si no, se emitirá un mensaje de error “el subtipo no existe”.

A continuación se muestra en la Figura 5.38 el código en LabVIEW [14] [15] [16] que implementa el servicio.

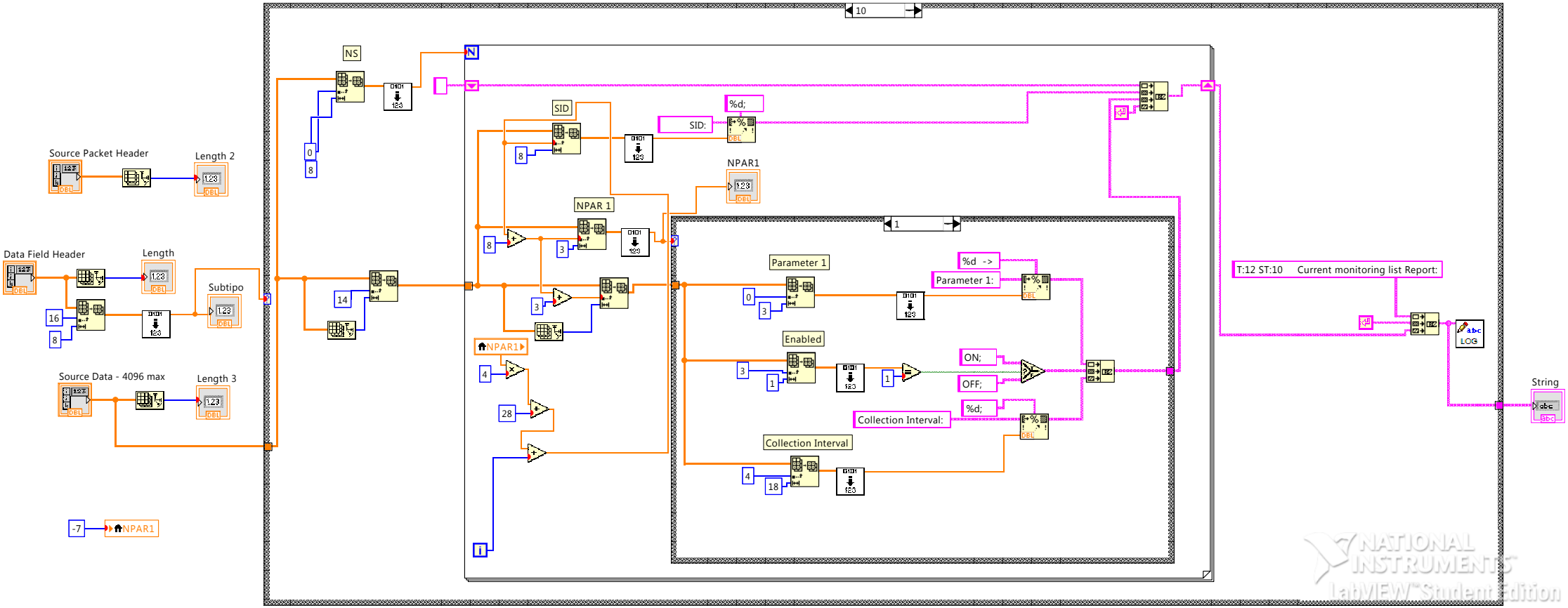


Figura 5.38: ***12 - monitorización a bordo.vi***

Current monitoring list report (12,10)

Este comando contiene la lista enviada por el satélite de los parámetros que están siendo monitorizados y la información de si estos están activos o no.

Al igual que pasaba con el servicio 3 (véase apartado 5.2.4), cada elemento del satélite está identificado mediante un SID (*Structure Identification*), y cada elemento contiene varios parámetros que se pueden monitorizar. Para este caso se han simulado los dos mismos elementos que en el servicio 3: el microprocesador (SID = 1), cuyo parámetro 1 es la temperatura y cuyo parámetro 2 es la frecuencia de funcionamiento; y los sensores (SID = 9), cuyo parámetro 1 es el sensor de temperatura y cuyo parámetro 2 es el de imágenes. Además de esta información también se envía el intervalo de recolección de estos datos de monitorización. La estructura del campo de datos del paquete de telemetría es:

NS	Spare	SID	NPAR1	Parameter N	Enabled	Collection Interval
8 Bits	6 Bits	8 Bits	3 Bits	3 Bits	1 Bit	18 Bits

Repetido NPAR1 veces

Repetido NS veces

El campo *NS* indica el número de elementos que se detallan en la lista. Después se añade un campo de relleno (*Spare*) para que la trama final contenga un número entero de octetos. A partir de ahí, la siguiente parte de la trama se repite *NS* veces y es la información de cada elemento. El campo *SID* es el identificador del elemento. El campo *NPAR1* informa acerca del número de parámetros que se encuentran en la lista (pueden ser 0, 1 ó 2), por lo que la siguiente parte que detalla la información de los parámetros se repetirá NPAR1 veces. El campo *Parameter N* identifica el parámetro en cuestión. El campo *Enabled* indica si el parámetro está activo o no. Y finalmente el campo *Collection Interval* contiene el intervalo con el que se recolectan los datos de monitorización.

Si se observa la Figura 5.38, se puede ver cómo se ha llevado a cabo la implementación de este servicio. En primer lugar se lee el parámetro *NS* para saber el número de elementos de la lista. Después el programa entra en un bucle tipo *For* que se repetirá *NS* veces y en el que se van leyendo los parámetros de la trama. Dependiendo del número de parámetros del elemento (indicado por el campo *NPAR1*), el programa llega a una estructura tipo *Case*, y actuará de distinta forma en función de si hay un parámetro, dos, o ninguno. En la Figura 5.38 se puede ver el caso en el que solo hay un parámetro. Para el caso en el que hay dos, el programa es exactamente el mismo solo que repetido dos veces; y para el caso en el que no hay ninguno, el programa emite el siguiente mensaje: “Parámetro no activo en el satélite”.

Una vez creada la lista con la información de los parámetros monitorizados se envía por la salida de texto para que el usuario la pueda ver.

5.2.10. 15 - test de conexión.vi

Este módulo implementa las funcionalidades del servicio 15, el encargado de realizar un test de conexión para ver si la comunicación entre el satélite y la estación de tierra es correcta. Cuenta solamente con un comando de telemetría que es el encargado de recibir la respuesta del satélite al test de conexión. Tiene las mismas tres entradas que los demás módulos (*Packet Header, Data Field Header* y *Source Data*) y consta de tres salidas: una de texto y dos indicadores luminosos. El símbolo del módulo es el siguiente:

Figura 5.39: Símbolo de *15 - test de conexión.vi*

Al igual que los módulos anteriores, lo primero que hace el programa es comprobar mediante una estructura tipo Case que el subtipo de servicio del paquete es el correcto. A continuación se muestra en la Figura 5.40 el código en LabVIEW [14] [15] [16] que implementa el programa.

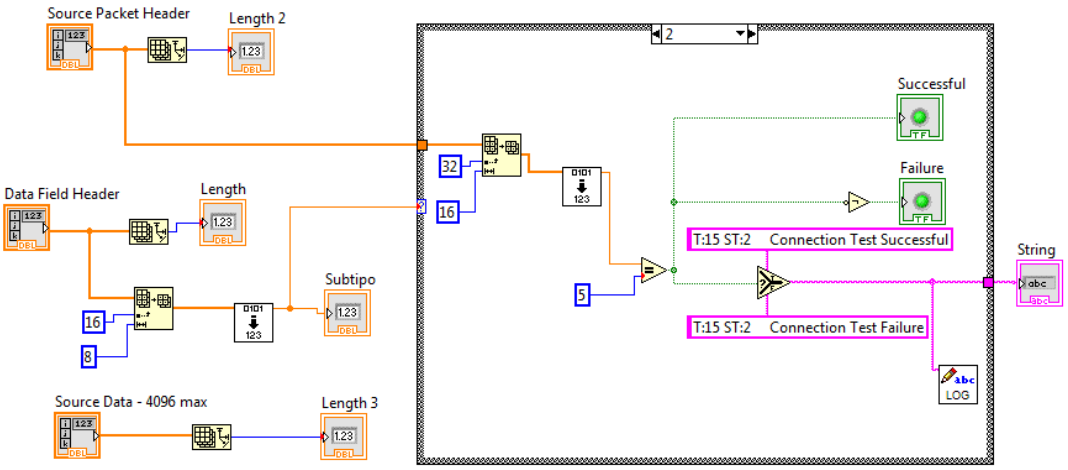


Figura 5.40: *15 - test de conexión.vi*

Connection test report (15,2)

Este comando de telemetría es enviado por el satélite a la estación de tierra como respuesta al test de conexión. Este comando de telemetría no cuenta con datos de aplicación, es decir, el campo *Source Data* va vacío. Por lo tanto el paquete solo está formado por las cabeceras.

Para comprobar que el paquete es correcto, se verifica que la longitud del paquete sea la adecuada. En este caso, como la longitud del campo *Packet Data Field* del paquete es de 6 octetos, si el paquete es correcto el campo *Length* debe contener un 5 (véase apartado 5.1.1). Y así es como se ha implementado este servicio en LabVIEW [14] [15] [16], como puede verse en la Figura 5.40. El programa comprueba que el campo *Length* contenga un 5 y después muestra un mensaje de texto en consecuencia y además enciende un indicador luminoso.

5.2.11. 32 - instrumento sensor temperatura.vi

Este módulo lleva a cabo las funciones del servicio 32, que se encarga de la gestión del sensor de temperatura del satélite. Este servicio dispone de tres comandos de telemetría encargados de interpretar los datos enviados desde el satélite. Consta de tres entradas (*Packet Header, Data Field Header* y *Source Data*) y de cuatro salidas: una salida de texto para informar al usuario, otra salida en forma de array de datos para poder representarlos posteriormente en una gráfica, y dos salidas de tipo indicador luminoso para informar al usuario. El símbolo del módulo es el siguiente:

Figura 5.41: Símbolo de *32 - instrumento sensor temperatura.vi*

Al igual que los demás módulos, lo primero que hace el programa es leer el subtipo de servicio al que pertenece el paquete recibido. Después el programa llega a una estructura tipo Case que tiene programada una acción diferente en función del subtipo de servicio del paquete. Si el subtipo no se corresponde con ninguno de los tres del servicio, el programa muestra un mensaje de error indicando que el subtipo de servicio es inválido.

A continuación se muestra en la Figura 5.42 el código de este módulo implementado en LabVIEW [14] [15] [16]. Y después se explica cómo se ha llevado a cabo cada uno de los subtipos de servicio.

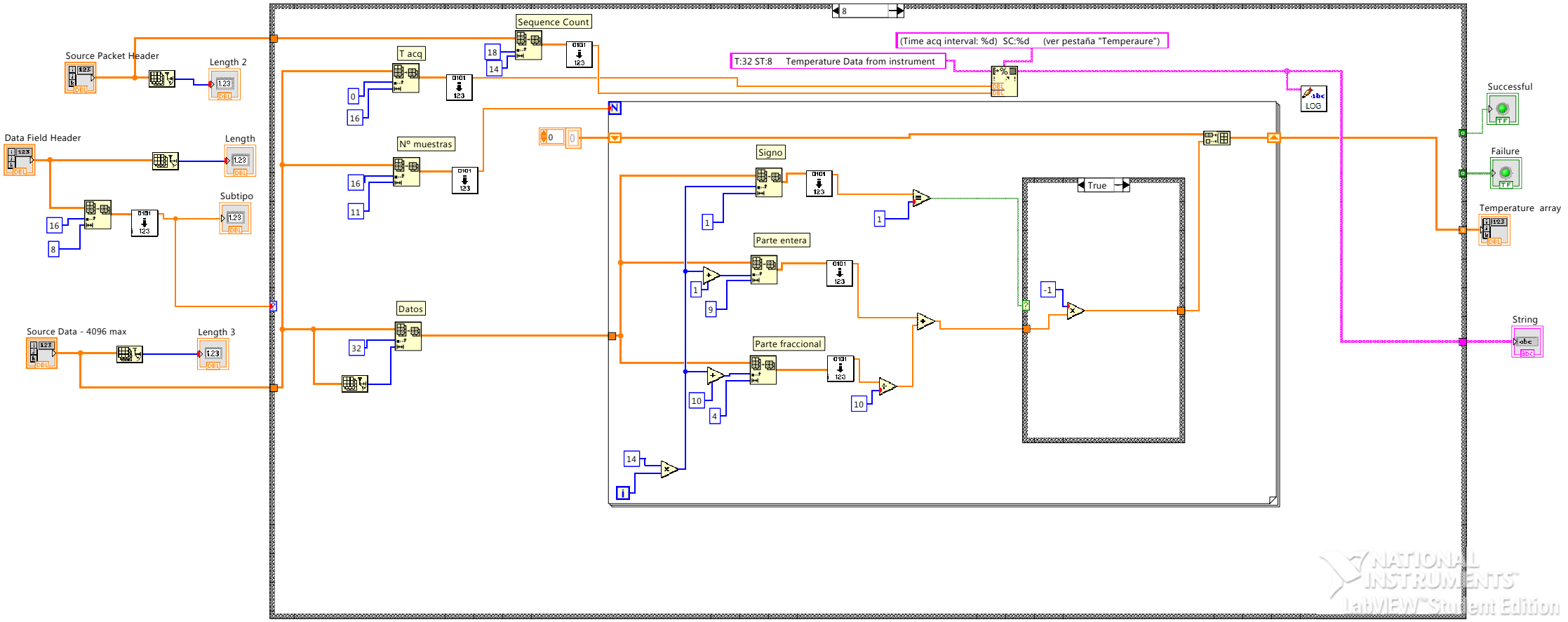


Figura 5.42: ***32 - instrumento sensor temperatura.vi***

Time acquisition interval report (32,5)

Este comando de telemetría se envía para informar al usuario de cuál es el intervalo de adquisición de los datos de temperatura del satélite. La estructura del campo de datos de este paquete de telemetría es la siguiente:

Time Acq. Interval
16 Bits

Como se puede ver, la trama tiene un único campo en el que se indica el intervalo de adquisición de datos. Como está formado por 16 bits, se podrá representar desde 0 a 65535 segundos que equivalen a 18,2 horas.

La implementación de esto se puede ver en la Figura 5.43. En ella se puede ver que el programa lo único que hace es leer el campo de datos, transformarlo a texto y enviarlo por la salida.

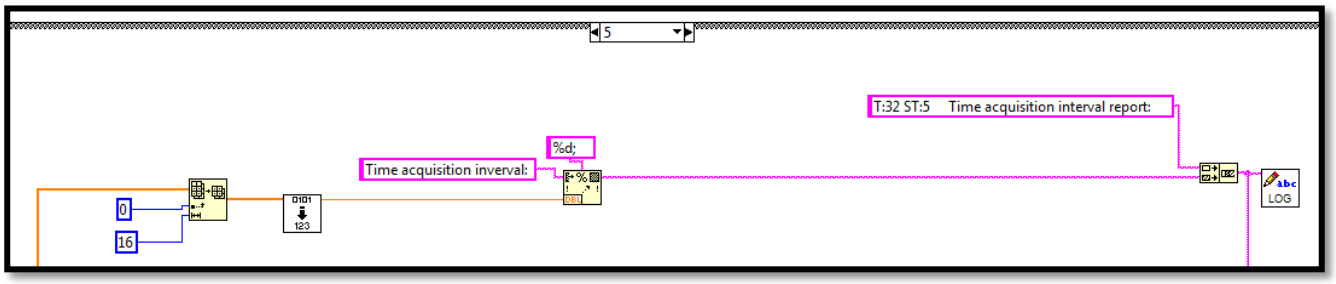


Figura 5.43: Implementación del servicio (32,5)

Data from instrument (32,8)

Con este comando de telemetría el satélite envía a la estación de tierra los datos obtenidos con el sensor de temperatura. El programa se encarga de leerlos y de representarlos en una gráfica. La estructura del campo de datos es la siguiente:

Time acq. Interval	Samples number	Spare	Data instrument
16 Bits	11 Bits	5 Bits	14 Bits x *Samples number*

El campo *Time acq. Interval* corresponde al intervalo de adquisición de los datos del sensor. El campo *Samples number* indica el número de muestras que se envía en el paquete de telemetría (cada muestra requiere de 14 bits). El campo *Spare* son 5 bits de relleno (ceros) que sirven para lograr que el paquete tenga un número entero de octetos. Finalmente, el campo *Data Instrument* contiene los datos de temperatura enviados por el satélite.

Por otro lado, cada muestra de temperatura enviada por el satélite se transforma a bits de la siguiente forma:

Sign	Decimal Sample Part	Fractional Sample Part
1 Bit	9 Bits	4 Bits

Como se puede observar, cada muestra de temperatura es un número fraccional y consta de 14 bits. El campo *Sign* indica el signo del número: si es positivo valdrá '0', y si es negativo valdrá '1'. El campo *Decimal Sample Part* contiene la parte decimal del número, mientras que el campo *Fractional Sample Part* contiene la parte fraccional. Con esta disposición de bits, se puede representar temperaturas desde los -511,15ºC hasta los +511,15ºC.

La implementación de este comando en LabVIEW [14] [15] [16] se puede ver a continuación en la Figura 5.43.

Como se puede observar, lo que hace el programa es leer el campo *Data Instrument* de 14 en 14 bits y va transformando los bits a números fraccionales. Esto lo hace mediante un bucle de tipo *For* que se repite *Samples Number* veces. Una vez obtenidos todos los datos de temperatura, el programa los envía por la salida en forma de array y posteriormente el módulo *Estación de tierra.vi* se encarga de representarlos.

Instrument test report (32,17)

Este comando de telemetría sirve para informar al usuario si el sensor de temperatura funciona correctamente o no. Si se reciben ocho ceros quiere decir que el sensor de temperatura funciona correctamente, mientras que si son ocho unos quiere decir que hay algún error. La estructura del campo de datos es:

Sensor Instrument Error
8 Bits

La implementación en LabVIEW [14] [15] [16] de este comando es relativamente sencilla, ya que solo hay que leer el campo *Sensor Instrument Error* para ver si contiene ocho unos u ocho ceros. Una vez hecho esto, se envía por la salida de texto un mensaje informando al usuario si el funcionamiento del sensor de temperatura es correcto o si, por el contrario, contiene algún error. Además, se enciende también un indicador luminoso en función de si hay error o no. El código del programa es el siguiente:

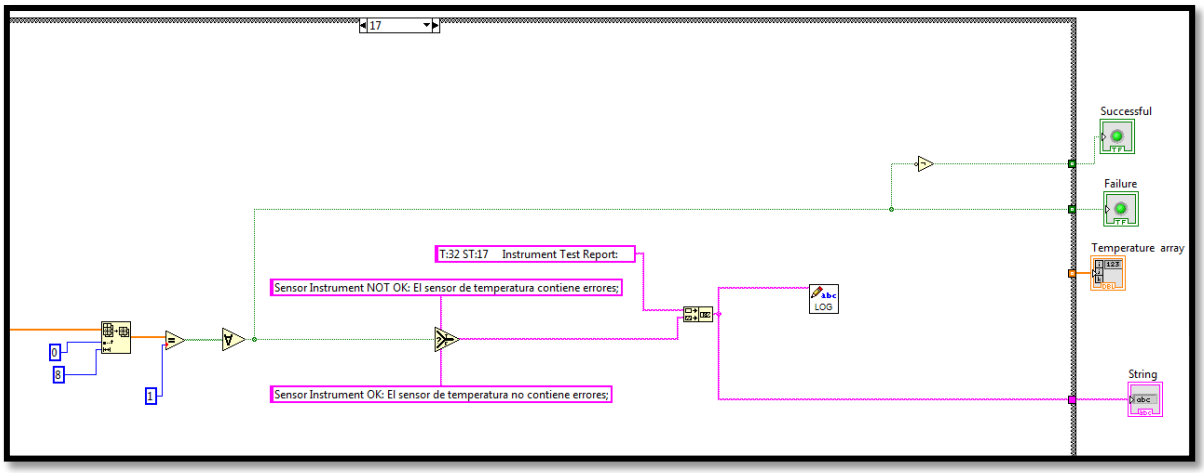


Figura 5.44: Implementación del servicio (32,17)

Transmission aborted (32,20)

Este comando de telemetría se envía cuando la transmisión de paquetes con los datos de temperatura del servicio (32,8) ha sido abortada. El campo de datos de este comando no contiene datos, así que el programa lo único que hace cuando le llega este comando es mostrar el mensaje "Transmission Aborted!!" y encender el indicador luminoso de fallo.

5.2.12. 64 - instrumento imágenes.vi

Este módulo implementa las funcionalidades del servicio número 64, que es el encargado de gestionar todo lo relacionado con el sensor de imágenes. Este módulo es bastante parecido al anterior. Cuenta también con tres comandos de telemetría, tres entradas (*Packet Header, Data Field Header* y *Source Data*) y con cuatro salidas (una de texto, un array de números y dos indicadores luminosos). El símbolo del módulo es el siguiente:

Figura 5.45: Símbolo de *64 - instrumento imágenes.vi*

Lo primero que realiza este módulo, al igual que los demás, es leer la cabecera *Data Field Header* que le llega como entrada para determinar a qué subtipo de servicio pertenece el paquete. Después de esto, el programa llega a una estructura tipo *Case* que tiene recogidos 5 casos distintos: cuatro que implementan los respectivos comandos de telemetría, y uno en el que se muestra un mensaje de error si el subtipo del paquete no se corresponde con ninguno de los tres anteriores. La estructura *Case* ejecutará uno u otro en función del número de subtipo que le llegue como entrada.

A continuación en la Figura 5.46 se muestra una captura del código en LabVIEW [14] [15] [16] que implementa el programa.

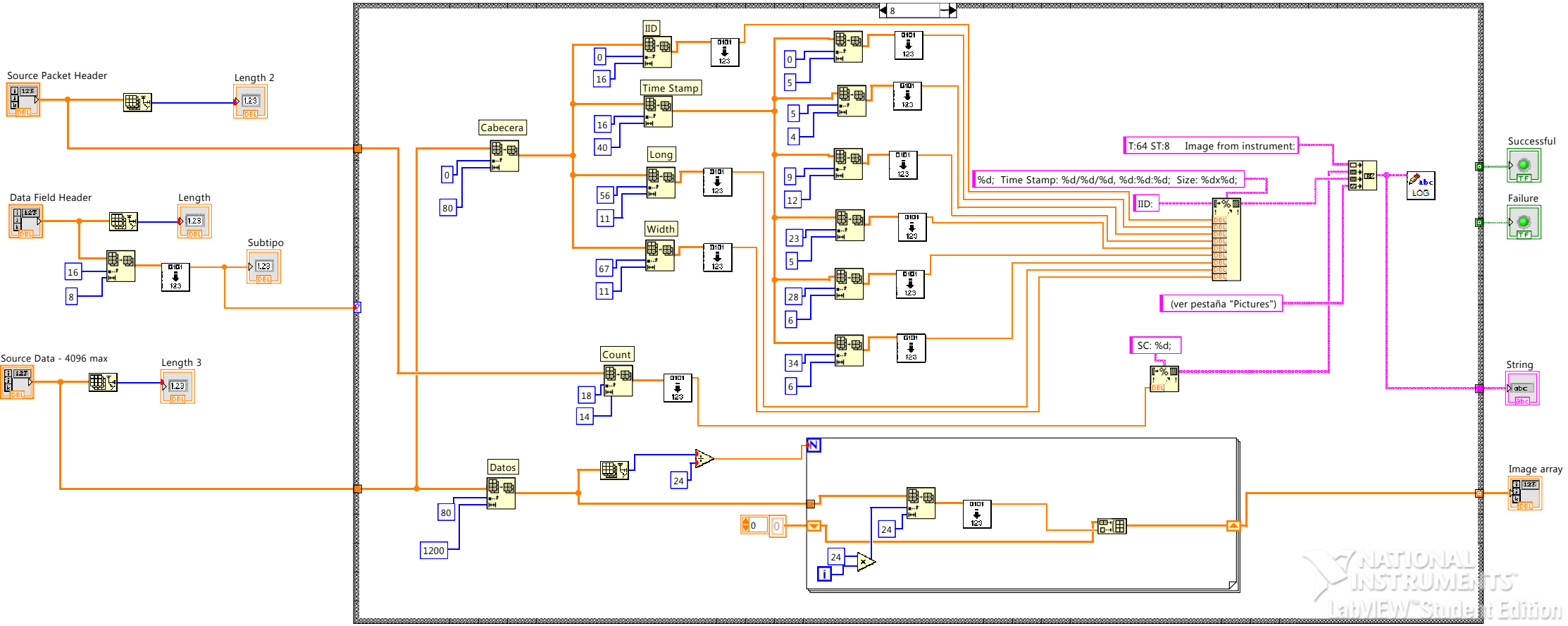


Figura 5.46: ***64 - instrumento imágenes.vi***

Time acquisition interval report (64,5)

Este comando de telemetría contiene la información relativa al intervalo de adquisición de imágenes por el sensor. El comando es exactamente igual que el (32,5) del apartado anterior (véase apartado 5.2.11). La estructura del campo de datos es la siguiente:

Time acq. Interval
16 Bits

El campo *Time acq. Interval* indica el intervalo de adquisición. Como consta de 16 bits, se consigue así representar intervalos desde 1 segundo hasta los 65535 segundos (18,2 horas). La implementación del comando es exactamente igual que la del servicio (32,5). (Véase Figura 5.42).

Data from instrument (64,8)

Este comando es probablemente uno de los más complejos de implementar en LabVIEW [14] [15] [16]. En este comando de telemetría se recibe una imagen codificada tomada por el sensor de imágenes del satélite. El problema reside en que las imágenes son ficheros de datos muy grandes y no se pueden enviar en un único paquete, por lo que hay que dividirlas en varios.

El módulo recibe los bits que forman la imagen y se encarga de transformarlos a valores en formato RGB. En el Anexo D se detalla cómo se codifica una imagen a bits. La estructura del campo de datos es la siguiente:

IID	Time Stamp	Length	Width	Spare	Data Instrument
16 Bits	40 Bits	11 Bits	11 Bits	2 Bits	Múltiplo de 8 Bits

El campo *IID* (*Image ID*) es el identificador de la imagen. El campo *Time Stamp* contiene la fecha y hora en la que la imagen es enviada; su formato es el mismo que en el servicio (9,3), y es el siguiente:

Day	Month	Year	Spare	Hour	Minute	Second
5 Bits	4 Bits	12 Bits	2 Bits	5 Bits	6 Bits	6 Bits

Los campos *Length* y *Width* indican el largo y ancho de la imagen respectivamente. El campo *Spare* contiene dos bits de relleno (‘00’) y el campo *Data Instrument* contiene los datos de la imagen.

La implementación de este servicio en LabVIEW [14] [15] [16] se puede ver una página atrás en la Figura 5.46.

Como se puede observar, lo primero que hace el programa es leer los campos que contienen información acerca de la imagen (*IID, Time Stamp, Length* y *Width*). Por otro lado y al mismo tiempo el programa también toma el campo de datos

(*Data Instrument*) y va leyendo los bits de 24 en 24 y transformándolos a números decimales que representan los coloren en el formato RGB. Estos valores los va guardando en un array y cuando termina los envía por la salida.

Como se comentó anteriormente, el problema de las imágenes es que son ficheros de datos muy grandes y no se pueden enviar en un único paquete, por lo que la imagen se recibe fragmentada en varios paquetes. El presente módulo solo se encarga de interpretar los valores de RGB de un paquete, ya que el módulo que se encarga de volver a unir la imagen es *Estación de tierra.vi*. En la Figura 5.47 se puede ver cómo une la imagen y la representa.

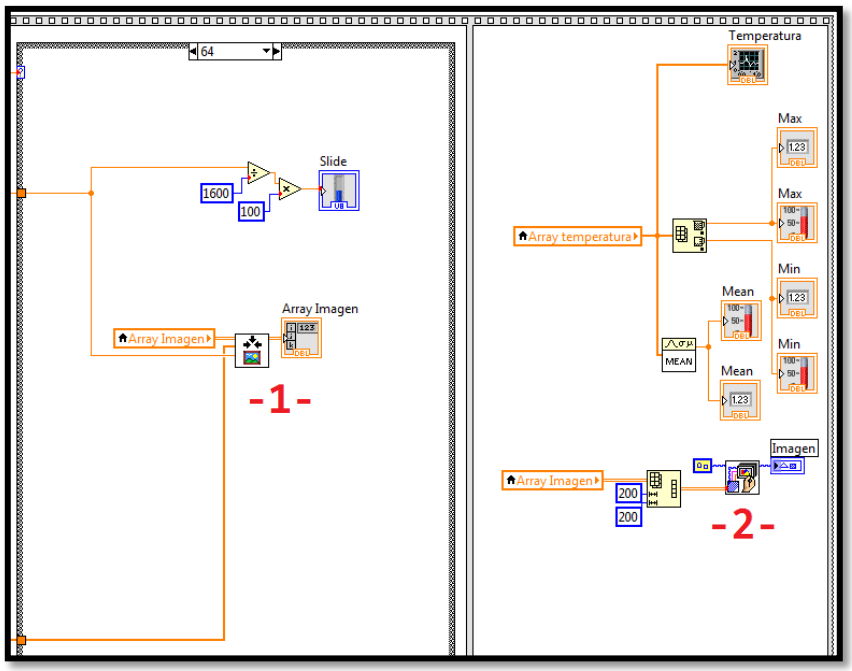


Figura 5.47: Implementación de la representación de imágenes

Como se puede ver en la imagen anterior, el programa concatena los datos nuevos recibidos de la imagen a los anteriores (véase Nº1 en la Figura 5.47) mediante el módulo auxiliar *Concatenar imagen.vi*, que se explicará a continuación; y después se representa la imagen (véase Nº2 en la Figura 5.47).

El módulo *Concatenar imagen.vi* es el encargado de volver a unir la imagen recibida. La forma de proceder de este módulo se puede ver en la página siguiente. El programa recibe como entradas el array bidimensional que representa la imagen y al que se deben concatenar los nuevos datos, un array unidimensional en el que se encuentran los nuevos datos a concatenar, y un parámetro *Contador* que sirve para saber en qué posición se han de introducir los nuevos datos.

El programa va leyendo los datos nuevos a concatenar y los va introduciendo en el array bidimensional de la imagen teniendo en cuenta el valor del parámetro *Contador*. Una vez concatenado el nuevo array, el array bidimensional de la imagen se envía como salida para que pueda ser representado.

Tanto el símbolo del módulo como el código en LabVIEW [14] [15] [16] del mismo se muestran a continuación en la Figura 4.48.

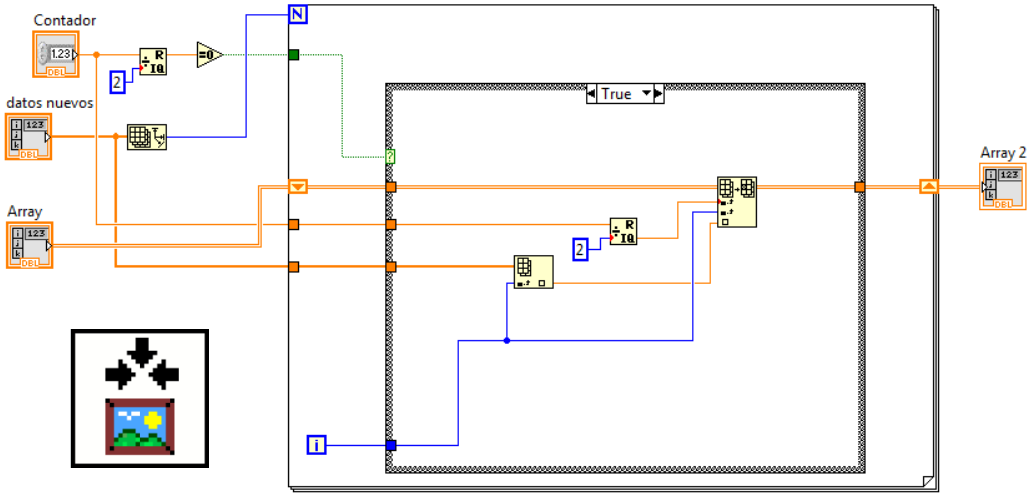


Figura 5.48: Símbolo y código de *Concatenar imagen.vi*

Instrument test report (64,17)

Este comando de telemetría es exactamente igual al (32,17) explicado anteriormente. La única diferencia es que ahora se envía en este comando el estado del sensor de imágenes y no el de temperatura. El comando indica si el sensor de imágenes funciona correctamente o si por el contrario hay algún error. La estructura del campo de datos de este comando es la siguiente:

Sensor Instrument Error
8 Bits

Si el campo *Sensor Instrument Error* contiene ocho ceros quiere decir que el funcionamiento del sensor es correcto. Por el contrario, si se reciben ocho unos quiere decir que hay algún error en el sensor de imágenes. El código del programa se muestra a continuación. En él, simplemente se comprueba el valor del campo anteriormente mencionado y se muestra un mensaje de texto para el usuario.

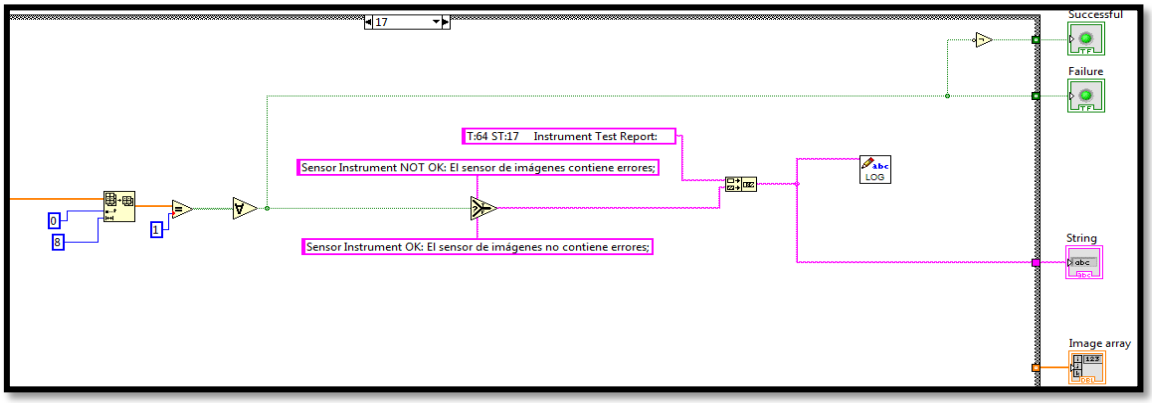


Figura 5.49: Implementación del servicio (64,17)

Transmission aborted (64,20)

Este comando de telemetría es exactamente igual que el del servicio de temperatura (32,20). Sirve para indicar al usuario que la transmisión de paquetes con los datos de la imagen ha sido abortada.

La programación de este subtipo de servicio en LabVIEW [14] [15] [16] es sencilla ya que el campo de datos del paquete (*Source Data*) no contiene datos. Por este motivo cuando se recibe este comando el programa lo único que hace es mostrar por pantalla el mensaje "Transmission aborted!!" y encender el indicador luminoso rojo para avisar de que se ha producido un fallo. Esto se muestra a continuación en la Figura 5.50.

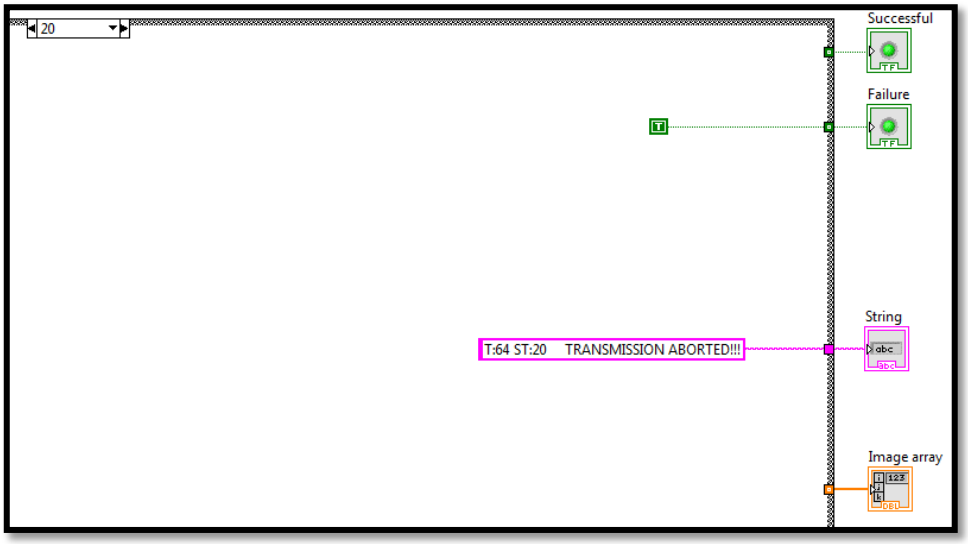


Figura 5.50: Implementación del servicio (64,20)

6. Pruebas y validación

En este capítulo se muestran los resultados de las pruebas realizadas para ver si se cumplen los objetivos marcados al principio de la memoria. También se mostrará cómo se cumplen esos objetivos, es decir, si son ampliamente satisfechos o no.

Recordando lo que se dijo en el apartado 1.2, los objetivos marcados para este proyecto eran: en primer lugar, conseguir simular la comunicación entre un satélite y una estación de tierra al juntar los tres proyectos realizados por el equipo del proyecto; en segundo lugar, en lo que respecta al presente proyecto, lograr interpretar correctamente la información enviada desde el satélite a la estación de tierra implementando las correspondientes funcionalidades; y en tercer lugar, hacer que el programa sea lo más fácil de usar y comprensible, de forma que los futuros estudiantes lo puedan utilizar para aprender.

A continuación se comprueba si se han cumplido estos tres objetivos marcados al comienzo del proyecto.

Se ha de aclarar que en este capítulo cuando se hable de “satélite”, en realidad se está refiriendo al transceptor NI USRP [7] – [10] que simula el satélite; y cuando se hable de “estación de tierra”, en realidad se refiere al transceptor NI USRP que simula la estación de tierra.

6.1. Transmisión / Recepción

En este apartado se prueba todo lo relativo a la comunicación entre el satélite y la estación de tierra, es decir, todo lo relativo a la transmisión y recepción de paquetes.

Durante la realización del proyecto, cuando se evaluaban el funcionamiento de la transmisión y recepción, se vio que no todos los paquetes que se enviaban desde el satélite llegaban a la estación de tierra. Unas veces era por fallos de los transceptores, y otras veces por fallos de los ordenadores.

Tras muchas comprobaciones, se determinó que las pérdidas de paquetes en la comunicación eran del 40%. Esto quiere decir que de cada 100 paquetes enviados, solo llegaban 60 y el resto se perdían.

Esta tasa de error era demasiado alta para una comunicación por satélite, por lo que se tuvieron que tomar medidas al respecto y mejorar el programa. Para

reducir el número de paquetes perdidos se pensaron dos posibles opciones, de las cuales una fue llevada a cabo.

En primer lugar, se pensó en realizar un sistema de transmisión de paquetes con espera y paquetes de confirmación (ACK). La idea era que la estación de tierra, cada vez que recibiera un paquete del satélite, le enviara a éste un paquete de confirmación para informarle de que lo había recibido. De este modo, si el satélite enviaba un paquete y pasado un tiempo (*timeout*) no recibía la respuesta de la estación de tierra, enviaría de nuevo el paquete.

El problema de esta solución era que requería que la estación de tierra pudiera recibir y transmitir mensajes dentro de un mismo programa. Es decir, que hubiera comunicación entre la transmisión y la recepción dentro de la estación de tierra. Sin embargo, como en este proyecto se han dividido las funciones de la estación de tierra en dos partes (telemetría y telecomandos), y cada alumno ha realizado una de estas partes, cada una de estas dos funciones está por lo tanto en un programa distinto. Por este motivo realizar la unión de ambas partes en una era muy difícil y laborioso.

No obstante, aunque realizar la unión entre las dos partes de la estación de tierra es difícil, no quiere decir que sea imposible. En este proyecto no se ha podido llevar a cabo principalmente por falta de tiempo, pero, como se comentará posteriormente en el capítulo de futuras líneas de trabajo, ésta es una de las posibles mejoras del proyecto que se podrían llevar a cabo.

Una vez descartada la anterior idea, se buscó otra solución que es la que se ha implementado finalmente.

Como no podía haber comunicación entre las partes de transmisión y recepción dentro de la estación de tierra, por lo que finalmente se optó es que el satélite enviara cada paquete repetido una serie de veces. Tras realizar muchas pruebas, se llegó a la conclusión de que el número óptimo de repetición de paquetes era 4. Por lo tanto, esto suponía que cada paquete enviado por el satélite se iba a enviar cuatro veces repetidas. De esta forma se aseguraba que el paquete llegara y se consiguió una tasa de fallo del 8% aproximadamente.

Sin embargo, esta solución suponía dos problemas: en primer lugar, la ralentización de la comunicación; y en segundo lugar, que, en el caso de que en una transmisión no se perdiera ningún paquete y llegaran los cuatro a la estación de tierra, ésta ejecutaría cuatro veces la función correspondiente.

En cuanto a lo primero, la ralentización de la comunicación es inevitable. Sin embargo, esta ralentización prácticamente no se aprecia, ya que la mayoría de los paquetes que manda el satélite son paquetes individuales y no se mandan de forma continuada. No obstante, sí que se aprecia cuando el satélite transmite los datos de temperatura o de imágenes en varios paquetes. Sobre todo éstos

últimos, llegando a tardar 10 minutos en transmitirse una imagen de 200x200 píxeles.

Y en cuanto a lo segundo, para solucionar que no se ejecute repetidas veces una función en la estación de tierra cuando llegue un paquete repetido del satélite, se ha modificado el programa para que lo primero que haga nada más recibir un paquete sea compararlo con el anterior para ver si es el mismo. Sin embargo, esto no interesa para los servicios (32,8) y (64,8) que son los que transmiten los datos de temperatura e imágenes (véase el apartado 5.2), ya que en estos servicios interesa que llegue la máxima cantidad de información posible (aunque sea repetida) para que se pueda reconstruir la imagen o la gráfica de la temperatura correctamente.

Para implementar esto en LabVIEW [14] [15] [16], simplemente se leen los campos *Sequence Count*, *Service Type* y *Service Subtype* (véase apartado 5.1) del paquete y solo si son distintos del anterior o si se trata del servicio (32,8) o (64,8) se ejecutará el programa. Esto se puede ver claramente en la Figura 6.1.

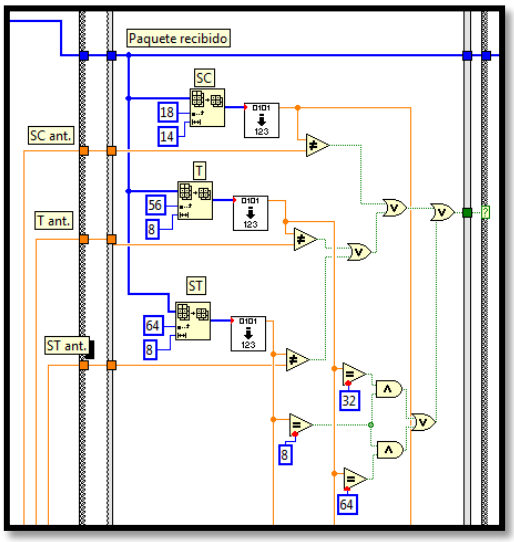


Figura 6.1: Implementación de la comprobación de la repetición de paquetes

Con esto se soluciona el problema de la pérdida de paquetes en la transmisión.

Tras realizarlo, se ha probado varias veces cada servicio implementado para estudiar en cuántas ocasiones han llegado los paquetes y en cuántas se han perdido, es decir, para calcular el porcentaje de paquetes transmitidos/perdidos. Esto se puede ver en la siguiente tabla. En ella se puede ver, para cada comando de telemetría que se ha probado, el número de paquetes que se han mandado con ese comando, y el número de paquetes que se han recibido. Para ver los datos relativos a los paquetes de telecomandos, consúltense las memorias de los otros alumnos encargados de implementarlos [2] [3].

	Comando	Paquetes Tx	Paquetes Rx	Porcentaje acierto (%)	Porcentaje fallo (%)
Servicio 1	TM(1,1)	315	309	98,10	1,90
	TM(1,2)	24	21	87,50	12,50
	TM(1,7)	315	305	96,83	3,17
	TM(1,8)	24	22	91,67	8,33
Servicio 3	TM(3,6)	36	32	88,89	11,11
	TM(3,9)	36	35	97,22	2,78
Servicio 5	TM(5,1)	60	57	95,00	5,00
	TM(5,2)	60	56	93,33	6,67
	TM(5,3)	60	57	95,00	5,00
	TM(5,4)	60	58	96,67	3,33
	TM(5,18)	31	24	77,42	22,58
	TM(5,20)	29	22	75,86	24,14
Servicio 6	TM(6,3)	28	26	92,86	7,14
	TM(6,5)	47	45	95,74	4,26
Servicio 9	TM(9,3)	56	54	96,43	3,57
Servicio 11	TM(11,8)	25	20	80,00	20,00
Servicio 12	TM(12,10)	30	28	93,33	6,67
Servicio 15	TM(15,2)	52	50	96,15	3,85
Servicio 32	TM(32,5)	26	24	92,31	7,69
	TM(32,8)	28	26	92,86	7,14
	TM(32,17)	44	42	95,45	4,55
	TM(32,20)	28	25	89,29	10,71
Servicio 64	TM(64,5)	28	26	92,86	7,14
	TM(64,8)	32000	31748	99,21	0,79
	TM(64,17)	43	41	95,35	4,65
	TM(64,20)	25	23	92,00	8,00

Tabla 6.1: Tabla de prueba de comandos de telemetría

Considerando la tabla anterior, se puede observar que se ha conseguido una tasa de transmisión media del 92,2% de los paquetes. Por lo tanto, se puede afirmar que este primer objetivo del proyecto se ha cumplido satisfactoriamente.

6.2. Funcionalidades

En este apartado se va a comprobar el correcto funcionamiento de las funcionalidades que se requerían para el proyecto y que han sido realizadas. Para más información sobre los servicios implementados, en el Anexo B se encuentra una tabla con todas las funciones de telemetría y telecomandos que se han implementado, y en el capítulo 5 de la memoria se describen estos comandos detenidamente.

A continuación se comprobará el funcionamiento de los servicios implementados.

En primer lugar, y en común a todos los servicios, cuando se recibe un paquete incorrecto, es decir que alguno de sus campos (*Version Number*, *Type*, *Data Field Header Flag*, *Service Type* o *Service Subtype*) contiene un valor erróneo, se muestra por pantalla un mensaje de error, como se puede observar a continuación en la Figura 6.2.

Figura 6.2: Fallos de paquete, tipo y subtipo incorrecto

Servicio 1: Verificación de telecomando

Este servicio va asociado a todos los demás servicios y, por cada paquete que la estación de tierra envía al satélite, éste enviará primero un paquete de "aceptación correcta" (1,1), luego el paquete que corresponda en función del servicio, y después un paquete de "ejecución correcta" (1,7). (Véase el capítulo 5.2). Esto se puede ver que funciona correctamente en la Figura 6.3.

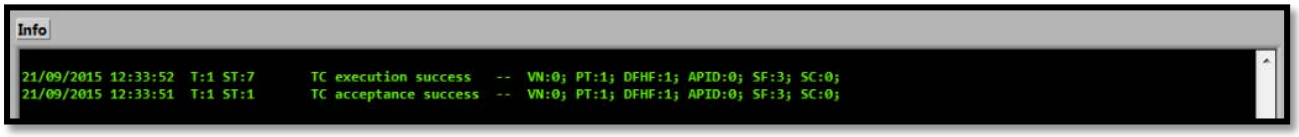


Figura 6.3: Prueba de aceptación y ejecución correcta

También cuando la aceptación o la ejecución han sido incorrectas se ha de mostrar un mensaje de error, como el que aparece en la Figura 6.4.

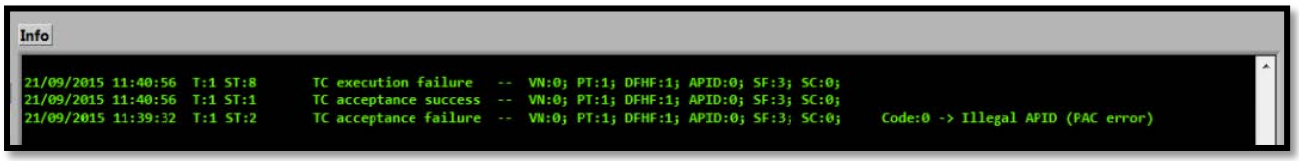


Figura 6.4: Prueba de aceptación y ejecución incorrecta

Servicio 3: Housekeeping y diagnóstico de datos

En este servicio, el comando (3,6) debe mostrar los datos de housekeeping enviados desde el satélite. (Véase el capítulo 5.2). En la pantalla "info" se informa al usuario a qué instrumento pertenecen los datos recibidos y se le indica que vea la pestaña "Graphics" para ver todos los datos enviados en forma de gráfica. Esto se puede ver en la Figura 6.5.

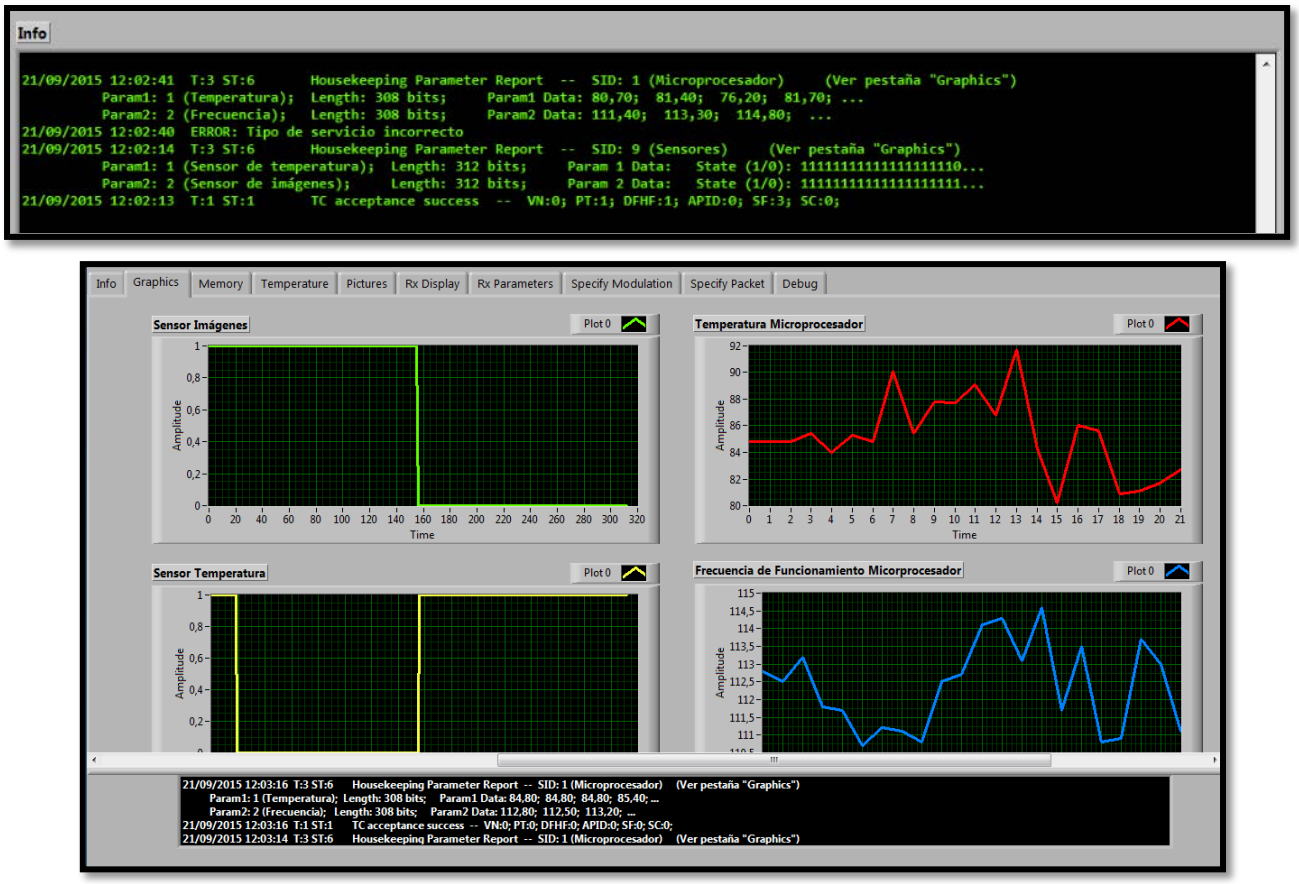


Figura 6.5: Prueba del servicio (3,6)

En cuanto al otro servicio, el (3,9), simplemente informa al usuario de que el período de recolección de datos ha sido actualizado, como se ve en la Figura 6.6.

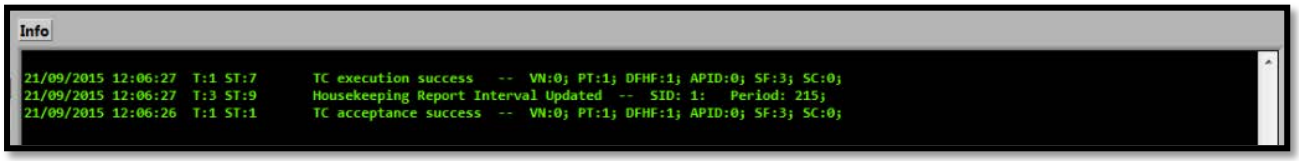


Figura 6.6: Prueba del servicio (3,9)

Servicio 5: Eventos

Los cuatro primeros subtipos de este servicio informan al usuario de los eventos producidos en el satélite, ya sean modificaciones o errores. (Véase el capítulo 5.2). En la Figura 6.7 se comprueba que esto funciona correctamente.

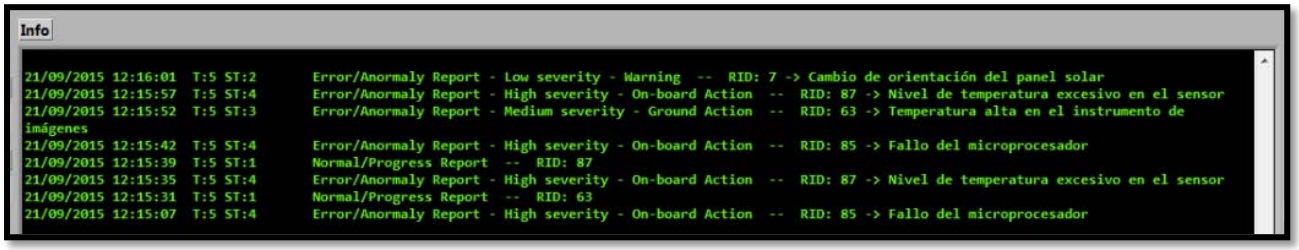


Figura 6.7: Prueba de los servicios (5,1) a (5,4)

En cuanto a los otros dos comandos de telemetría, el (5,18) y el (5,20), muestran la lista de los parámetros que tienen activa la generación de eventos y los que no, respectivamente. Nótese que estas dos listas son complementarias, es decir, que los parámetros que están en una de las listas no está en la otra y viceversa. Su correcta ejecución se puede ver en la Figura 6.8.

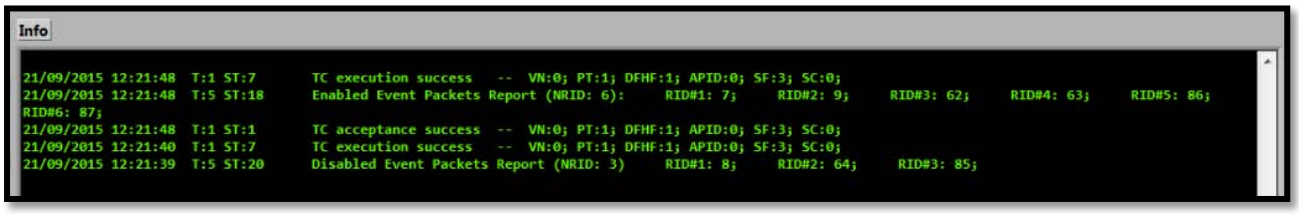


Figura 6.8: Prueba de los servicios (5,18) y (5,20)

Servicio 6: Gestión de la memoria

El comando de telemetría (6,3) contiene una porción de memoria del satélite que se ha solicitado previamente desde la estación de tierra. (Véase el capítulo 5.2). Cuando se recibe este comando se informa al usuario en la pantalla “info” de la dirección de comienzo del bloque de memoria y se le indica que consulte la pestaña “Memory” para ver esta porción de memoria. Todo esto se puede ver cómo funciona correctamente en la Figura 6.9.

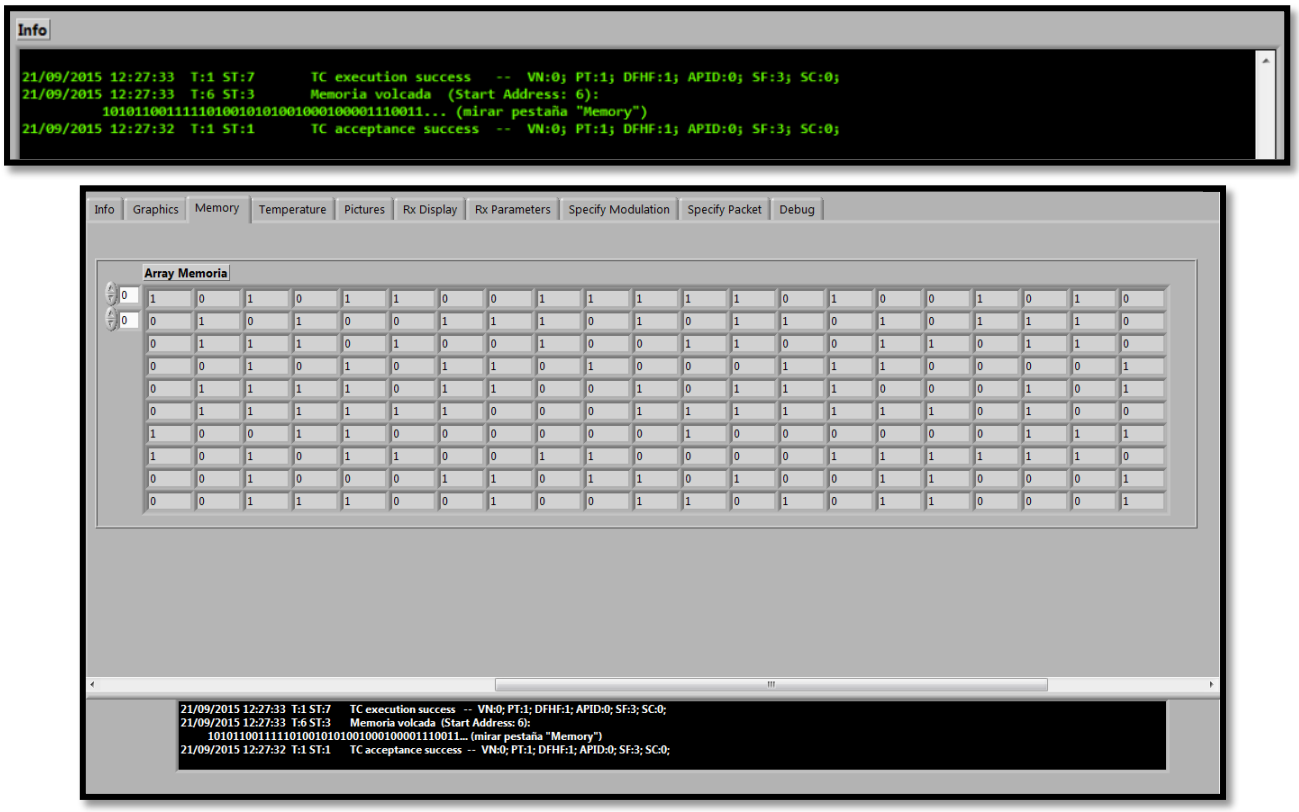


Figura 6.9: Prueba del servicio (6,3)

En cuanto al servicio (6,5), simplemente se informa al usuario si la memoria del satélite contiene errores o no. Ambos casos se pueden ver en la Figura 6.10.

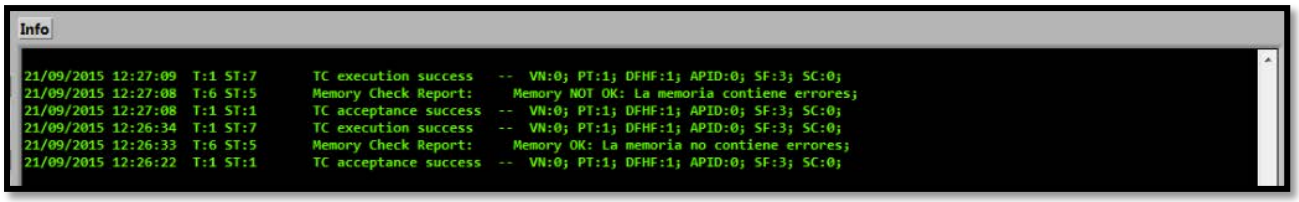


Figura 6.10: Prueba del servicio (6,5)

Servicio 9: Gestión del tiempo

En este servicio solo hay un comando de telemetría, el (9,3). En él se envía la fecha y hora que el satélite tiene configurado. (Véase el capítulo 5.2). Como se puede ver en la Figura 6.11, en la pantalla "Info" se muestra la hora enviada desde el satélite.

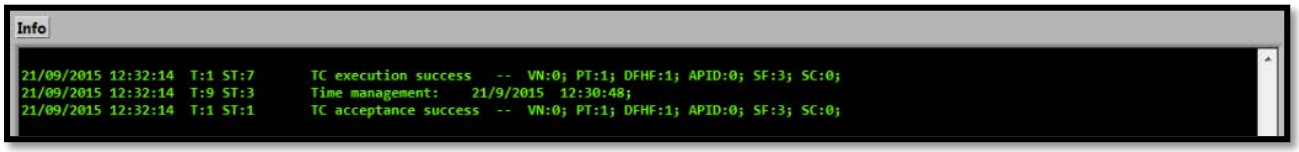


Figura 6.11: Prueba del servicio (9,3)

Servicio 11: Gestión del planificador de telecomandos

Este servicio solo posee un comando de telemetría, el (11,8), en el que el satélite envía la lista de funciones de telecomandos que tiene programadas. (Véase el capítulo 5.2). En la pantalla "info" se muestra al usuario el ID del telecomando, el tipo y subtipo al que pertenece, y la hora a la que está programada su ejecución. Esto se puede ver que funciona correctamente en la Figura 6.12.

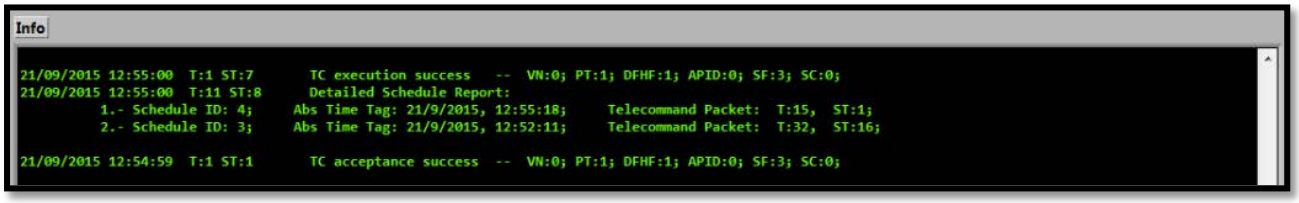


Figura 6.12: Prueba del servicio (11,8)

Servicio 12: Monitorización de a bordo

Este servicio se encarga de la monitorización de los parámetros de a bordo. El comando de telemetría (12,10) –el único que hay– sirve para enviar a la estación de tierra una lista de los parámetros que están siendo monitorizados e indicar cuáles están activos y cuáles no. (Véase el capítulo 5.2). El funcionamiento de este servicio se puede comprobar que es correcto en la Figura 6.13.

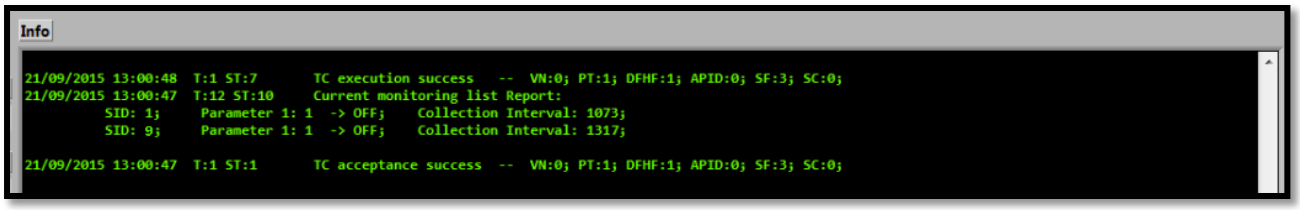


Figura 6.13: Prueba del servicio (12,10)

Servicio 15: Test de conexión

Este servicio sirve para comprobar que existe comunicación entre el satélite y la estación de tierra. (Véase el capítulo 5.2). El comando (15,2) es la respuesta del satélite a la solicitud del test de conexión. Si el test de conexión ha sido satisfactorio, se mostrará un mensaje de éxito en la pantalla; si por el contrario el test ha sido fallido, se mostrará un mensaje de error. Ambos casos se pueden ver en la Figura 6.14.

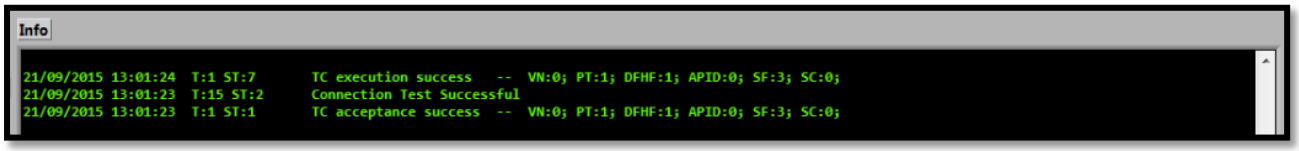


Figura 6.14: Prueba del servicio (15,2)

Servicio 32: Instrumento sensor de temperatura

Los servicios (32,5) y (32,17) sirven, respectivamente, para informar al usuario de que el tiempo de adquisición de datos se ha actualizado, y para mostrar si el sensor funciona correctamente. (Véase el capítulo 5.2). En la Figura 6.15 se muestra el correcto funcionamiento de los dos servicios.

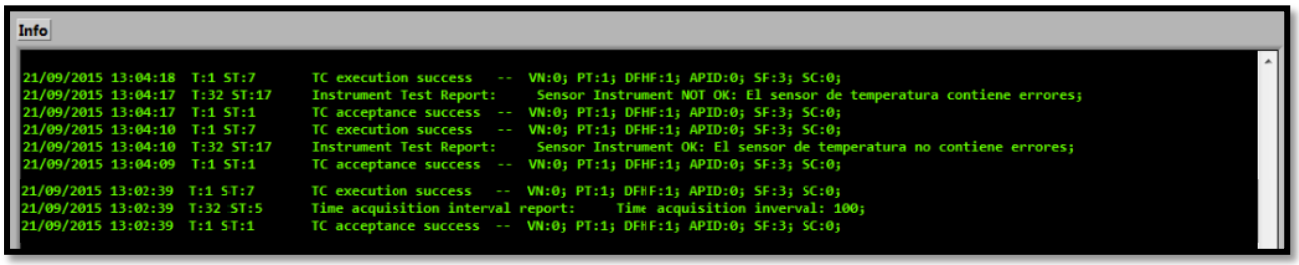


Figura 6.15: Prueba de los servicios (32,5) y (32,17)

Por otro lado, el servicio (32,8) es en el que se envían los datos de temperatura que ha recogido el satélite. En la pantalla "info" se indica al usuario que consulte la pestaña "Temperature" para que pueda ver los datos en forma de gráfica. En esta pestaña se puede ver la gráfica que representa los datos de temperatura, una barra de progreso que indica el porcentaje de datos que se ha recibido, y tres termómetros que muestran la temperatura máxima, mínima y media registrada, respectivamente. Un ejemplo de esto se puede ver en la Figura 6.16, donde se ve que funciona correctamente.

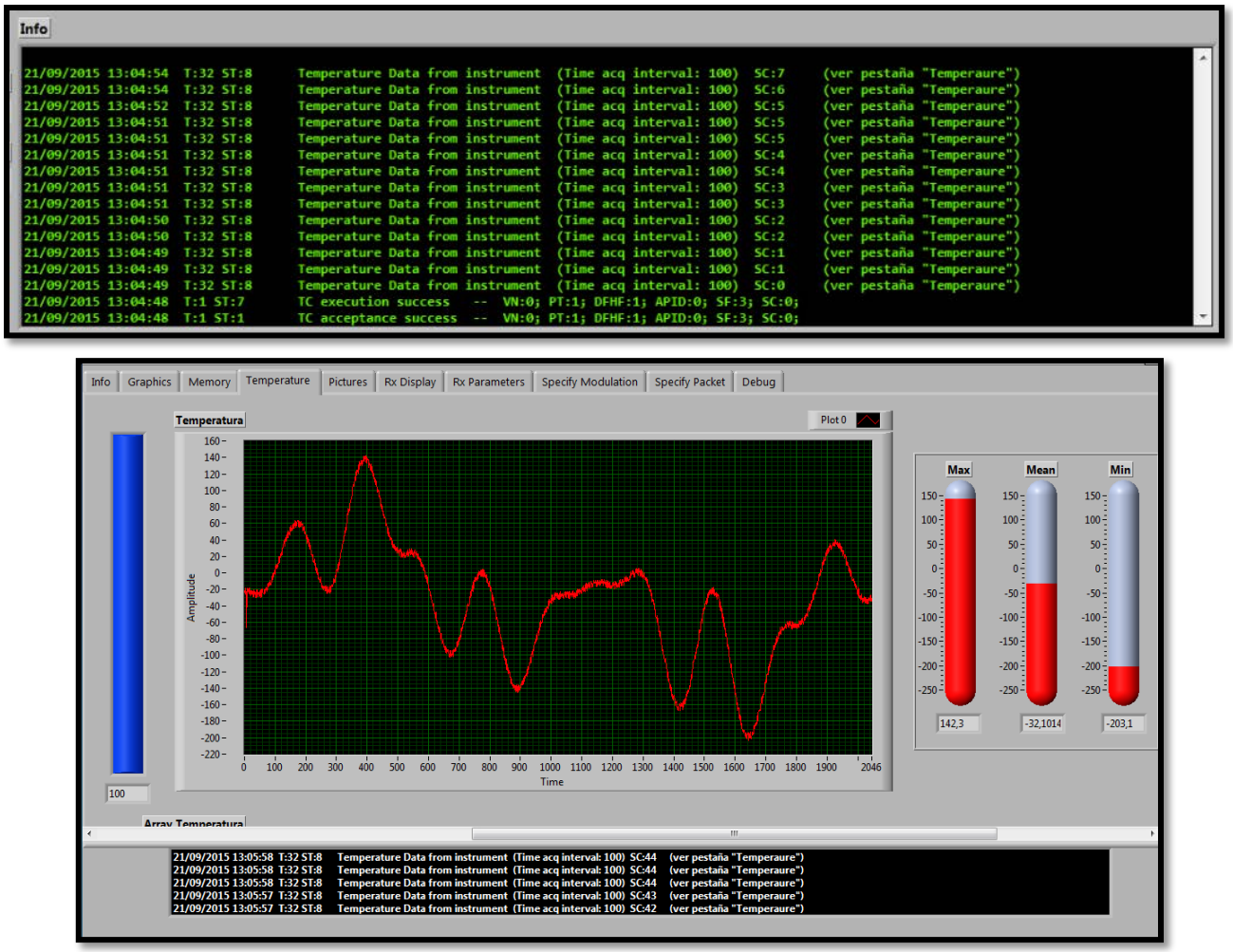


Figura 6.16: Prueba del servicio (32,8)

Además de esto, si la transmisión de los datos de temperatura ha sido abortada, llegará el comando de telemetría (32,20) para informar al usuario, como se puede ver en la Figura 6.17.

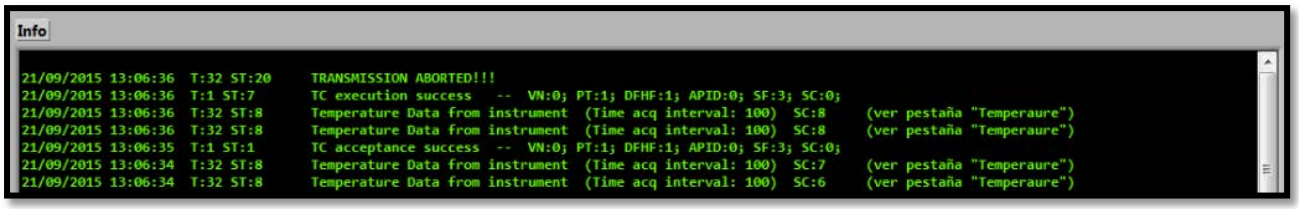


Figura 6.17: Prueba del servicio (32,20)

Servicio 64: Instrumento de imágenes

Este servicio es muy similar al anterior, solo que ahora se trata de gestionar imágenes en lugar de datos de temperatura. Al igual que antes, los servicios (64,5) y (64, 17) sirven para informar al usuario del tiempo de adquisición y del estado del sensor, respectivamente. (Véase el capítulo 5.2). El correcto funcionamiento de estos dos servicios se muestra en la Figura 6.18.

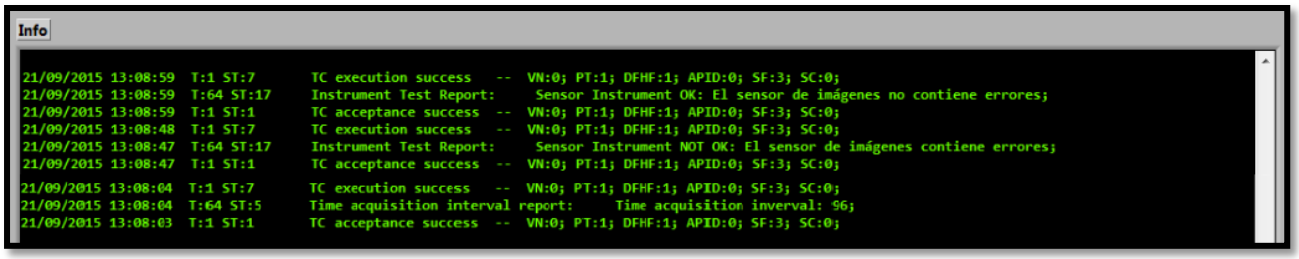


Figura 6.18: Prueba de los servicios (64,5) y (64,17)

Al igual que en el servicio de temperatura, el servicio (64,8) es en el que se transmiten los datos de la imagen. En la pantalla "info" se indica al usuario que vaya a la pestaña "Pictures" para ver la imagen. En esta pestaña se puede ver la imagen enviada, una muestra del array de datos de la imagen, una barra de progreso que indica el porcentaje de la imagen que se ha recibido, y un botón para guardar la imagen en el ordenador. Cuando el usuario acciona el botón se abre una ventana emergente para elegir dónde guardar la imagen. Todo esto se puede ver que funciona correctamente en la Figura 6.19.

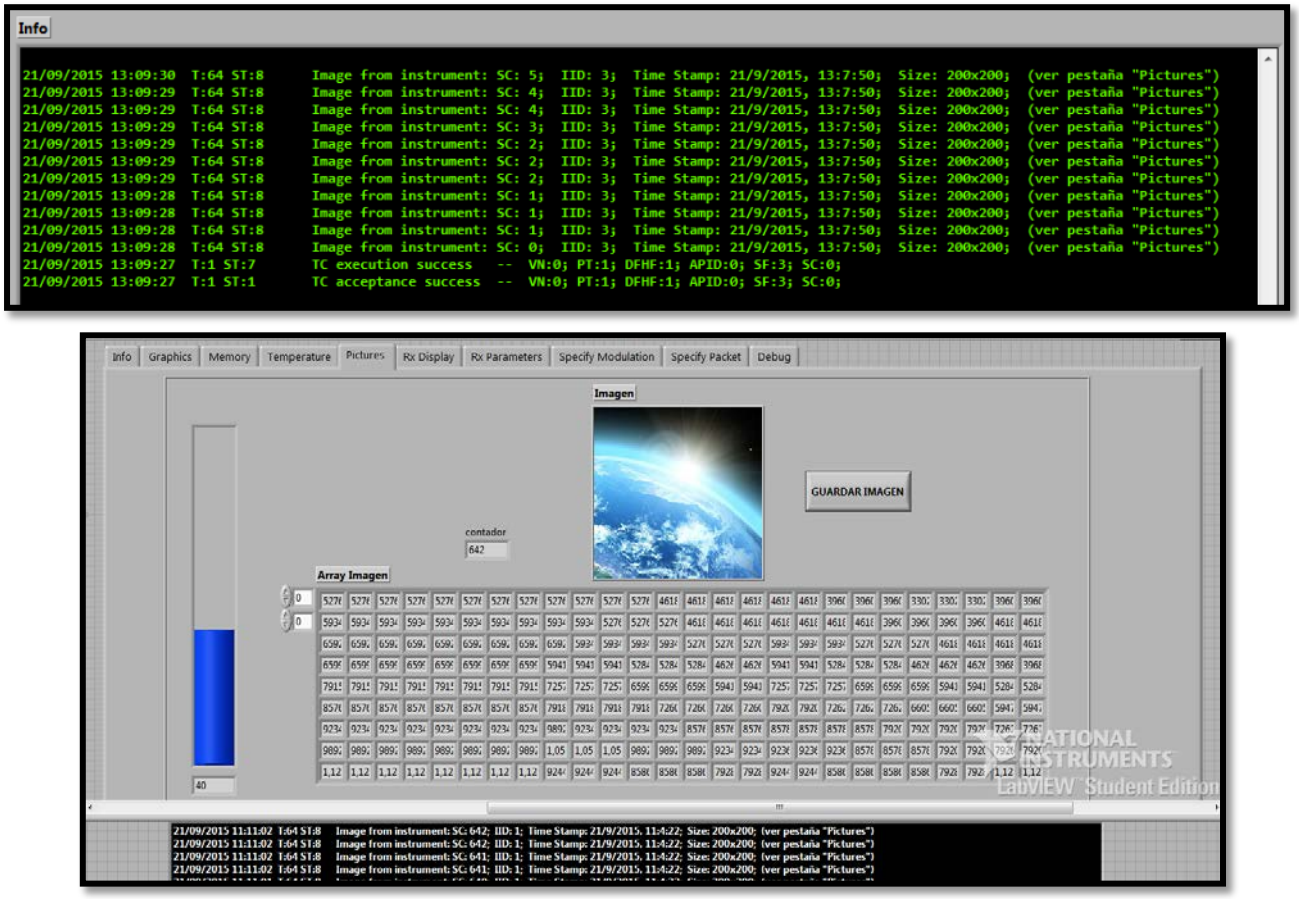


Figura 6.19: Prueba del servicio (64,8)

Al igual que antes, si la transmisión de los datos es abortada, llegará un paquete de telemetría con el comando (64,20) para indicar al usuario que se ha interrumpido la transmisión. Esto se puede ver en la Figura 6.20.

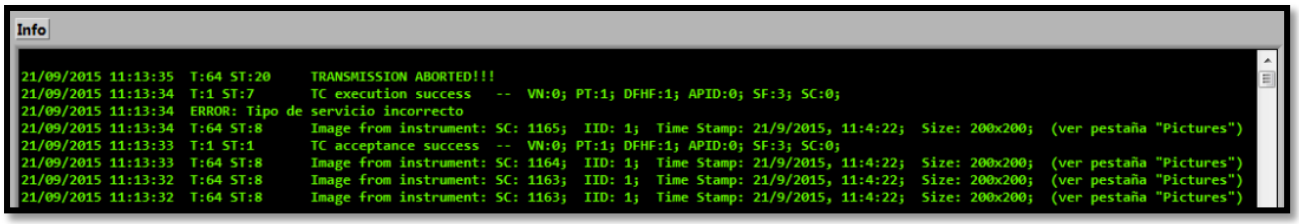


Figura 6.20: Prueba del servicio (64,20)

En este apartado de Funcionalidades se ha comprobado el funcionamiento de todos los servicios implementados en el proyecto y se ha visto que funcionan todos correctamente. Por lo tanto, se puede afirmar que este objetivo del proyecto ha sido cumplido satisfactoriamente.

6.3. Facilidad de uso

El tercer y último objetivo que se buscaba en este proyecto era realizar un simulador de una comunicación por satélite que fuera de fácil manejo y lo más sencillo posible. Se planteó así porque el proyecto está pensado para que los futuros estudiantes lo puedan utilizar para comprender mejor cómo funciona una comunicación por satélite. En este apartado se comprobará si estos objetivos se han cumplido o no.

En primer lugar, en cuanto a la interfaz, se ha buscado que sea lo más sencilla e intuitiva posible. Ésta se muestra en la Figura 6.21.

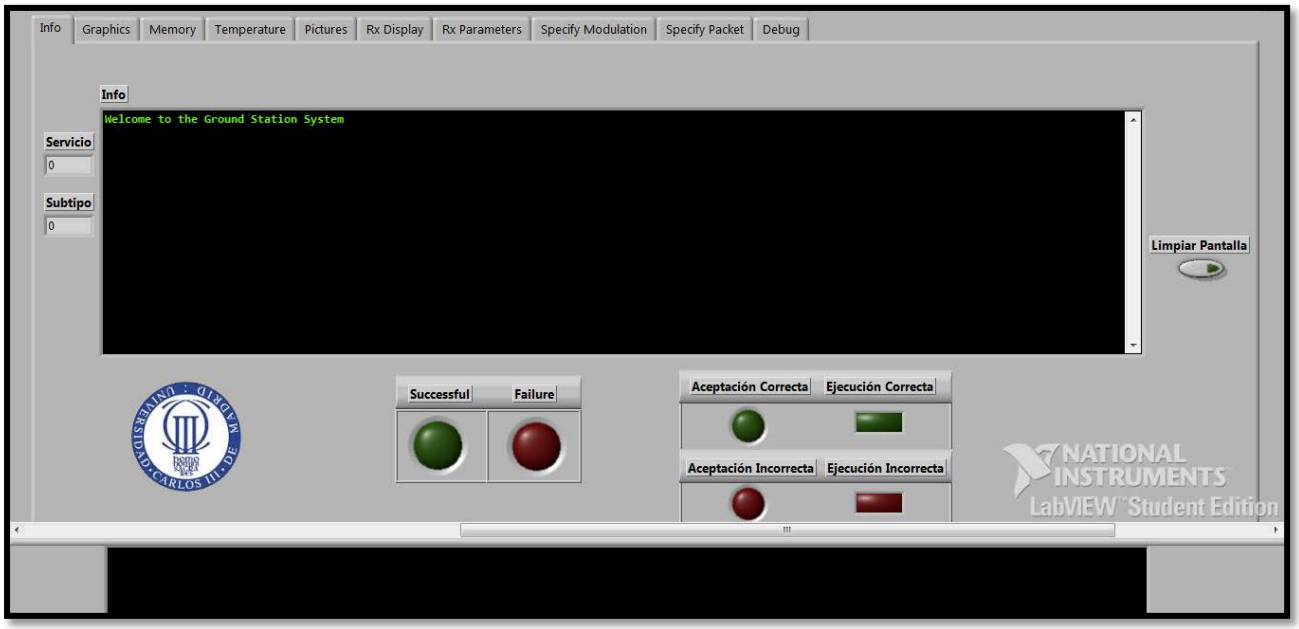


Figura 6.21: Interfaz del programa

Como se puede ver en la Figura 6.21, la interfaz del programa es muy sencilla y a la vez eficaz. La pestaña principal del programa (*Info*) cuenta con una pantalla donde se muestra todo lo que ocurre en cada momento de la comunicación.

También se ha implementado un botón situado a la derecha de la pantalla para borrar la pantalla cuando el usuario lo estime oportuno. Sin embargo, toda la información recibida por la estación de tierra es almacenada en un registro (*log*), y el hecho de borrar la pantalla no implica que se borre la información del registro.

Además de la pantalla principal (*Info*) del programa, hay otras pestañas auxiliares: unas que sirven para configurar los parámetros de la comunicación, y otras que permiten al usuario observar información sobre otros aspectos de la comunicación u otros datos que por su formato no podían ser mostrados por la pantalla.

Las pestañas que sirven para configurar la comunicación son: *Rx Parameters*, donde se configura la dirección IP del transceptor NI USRP [7] – [10] y la frecuencia a la que va a funcionar; *Specify Modulation*, que permite elegir el tipo de modulación deseada y los parámetros del filtro; y *Specify Packet*, donde se configura la longitud de los mensajes y el número de bits de guarda y sincronización.

En cuanto a las pestañas auxiliares que complementan a la pestaña principal, éstas son: *Graphics*, que permite ver los datos relativos al servicio 3 (Housekeeping) en forma de gráfica; *Memory*, en la que se muestra la porción de memoria enviada desde el satélite; *Temperature*, que permite ver en forma de gráfica los datos de temperatura enviados en el servicio 32; *Pictures*, donde se muestra la imagen recibida en la estación de tierra a través del servicio 64; *Rx Display*, que permite ver la señal recibida, la constelación formada y los bits recibidos; y *Debug*, donde se puede ver la relación señal a ruido de la señal recibida.

El manejo de todas estas pestañas es muy sencillo. De todas formas, en el Anexo E de la memoria se adjunta un manual del programa donde se explica cómo utilizar el simulador. De esta forma le será más fácil al usuario entender y usar el programa.

Además de esto, cabe destacar también que todos los diagramas de bloque que se han diseñado son de código abierto, es decir, que cualquier futuro alumno podrá ver cómo se han implementado las funciones realizadas en el proyecto. Esto permitirá a los alumnos venideros tanto comprender cómo se ha hecho el programa, como poder modificarlo y añadir nuevas mejoras.

Por lo tanto, y con todo lo mencionado anteriormente, se puede ver que el proyecto que se ha desarrollado está totalmente orientado a la educación y que se ha hecho buscando que sea lo más sencillo de entender posible, lo más manejable posible y con vistas a futuras mejoras.

Por estos motivos, se puede afirmar que el proyecto ha cumplido satisfactoriamente todos los objetivos marcados al comienzo del mismo.

7. Presupuesto

En este capítulo se detalla el presupuesto necesario para llevar a cabo el proyecto. Se contará tanto el coste de personal como el coste de material, licencias, etc.

En primer lugar, en cuanto a los costes de personal, como se mencionó en el apartado 1.5 dedicado al cronograma, para llevar a cabo el proyecto son necesarios 188 días de trabajo. Este dato se calculó realizando una estimación media de dedicación al proyecto de unas 4 horas al día de media.

También hay que tener en cuenta que para desarrollar el proyecto completo (el satélite y la estación de tierra) han sido necesarios tres alumnos graduados y un tutor responsable del proyecto. Se calcula entonces que los alumnos han trabajado un total de 752 horas cada uno y el tutor un total de 75 horas (aproximadamente el 10%). El coste por hora del personal y el coste total del mismo se detallan a continuación en la Figura 7.1.

Personal	Tiempo de trabajo (horas)	Coste por hora (€/h)	Coste (€)
Tutor	75	80	6.000
Alumno graduado	752	40	30.080
Alumno graduado	752	40	30.080
Alumno graduado	752	40	30.080
		TOTAL	**96.240 €**

Tabla 7.1: Costes de personal

En segundo lugar, en cuanto a los materiales necesarios, han sido necesarios varios equipos y licencias para llevar a cabo el proyecto. Éstos han sido: 3 transceptores NI USRP [7] – [10], 1 ordenador PC, 2 ordenadores portátiles y la licencia del programa LabVIEW [14] [15] [16].

Para calcular los costes de estos materiales es necesario calcular las amortizaciones producidas. Esto se ha hecho aplicando los porcentajes de pérdida de valor establecidos por la Agencia Tributaria [39]. La fórmula para calcularlo es por lo tanto:

$$Coste = Coef.amortización \times \frac{Nº\ meses\ de\ utilización}{12} \times Precio\ del\ material$$

A continuación se muestran en la Figura 7.2 todos los datos mencionados anteriormente y el coste total de los materiales.

Material	Precio unidad (€)	Nº meses de utilización	Coef. Amortización [39]	Coste (€)
3 x NI USRP	2.870	3,5	26%	652,92
1 x Ordenador	850	6	26%	110,5
2 x Ord. Portátil	640	6	26%	166,4
1 x Licencia LabVIEW	50.000	3,5	26%	3.791,67
			TOTAL	**4.721,49 €**

Tabla 7.2: Costes de materiales

Para hallar el coste total del proyecto hay que sumar al coste de personal y de los materiales los costes indirectos (que suponen el 20% de los directos) y el IVA (21%). Esto se muestra a continuación en la Figura 7.3.

	Coste (€)
Coste de personal	96.240
Coste de materiales	4.721,49
Costes indirectos (20%)	20.192,3
Subtotal	121.153,79
+ IVA (21%)	25.442,3
TOTAL	**146.596,09 €**

Tabla 7.3: Costes totales

Con todo lo anterior, el coste total del proyecto asciende, por tanto, a la cantidad de 146.596,09 €.

8. Conclusiones y futuras líneas de trabajo

Para finalizar la memoria, en este último capítulo se presentan, en primer lugar, las conclusiones generales del proyecto y, en segundo lugar, las futuras líneas de trabajo por las que se podría continuar el proyecto.

8.1. Conclusiones

Como se comentó anteriormente, este proyecto surgió con motivo de la falta en la Universidad Carlos III de Madrid de un simulador de una comunicación por satélite. La idea fue propuesta por el tutor del proyecto y ha sido llevada a cabo por tres alumnos de la Universidad.

Cuando se comenzó el proyecto, en la fase de planificación, se plantearon una serie de objetivos que el proyecto debía lograr. Estos eran: en primer lugar, y como objetivo común de los tres alumnos del proyecto, lograr realizar una simulación de una comunicación por satélite implementada en una estructura definida por software (SDR); en segundo lugar, y como objetivo del presente proyecto, lograr implementar las funcionalidades de telemetría de una estación de tierra perteneciente a una comunicación por satélite; y en tercer lugar, hacer un programa sencillo, de cómodo manejo y de fácil comprensión.

Además de estos objetivos, también se buscaba profundizar en los temas estudiados en el grado (sobre todo lo relativo a satélites), investigar sobre las tecnologías que se han utilizado (SDR, LabVIEW…) y ampliar de este modo los conocimientos sobre este campo de las telecomunicaciones.

Relacionado con esto último, antes de comenzar la implementación del proyecto, hubo que hacer un trabajo de investigación, lo que viene recogido en los capítulos 2, 3 y 4 de la memoria.

Primero se investigó sobre las tecnologías que se emplean en el proyecto. Éstas eran las comunicaciones por satélite [1] y la tecnología SDR [4] [5] [6].

En cuanto a las primeras, se buscó información sobre qué son las comunicaciones por satélite [1], sobre su historia, y sobre los dos elementos principales de la comunicación por satélite: el satélite y la estación de tierra. También se definieron los conceptos de telemetría y telecomandos, ampliamente utilizados en este proyecto.

Y en cuanto a la tecnología SDR (Software Defined Radio) [4] [5] [6], también se realizó un proceso de investigación acerca de lo que es, su evolución, y de cómo y cuánto se utiliza actualmente esta tecnología.

En segundo lugar, se investigó también sobre el entorno de trabajo del proyecto. Aquí se buscó información del funcionamiento de las dos principales herramientas utilizadas para llevar el proyecto a cabo: el transceptor NI USRP 2920 y el entorno de programación LabVIEW.

En cuanto al transceptor NI USRP [7] – [10], se buscó información acerca de las características de funcionamiento del mismo. Y en cuanto al entorno de programación LabVIEW [14] [15] [16], se consultaron manuales sobre la forma de utilización del mismo y sobre cómo realizar programas con él.

Y en tercer lugar, se investigó sobre otros proyectos similares que existieran ya, tanto en el mercado, como en las universidades y se realizó una comparación entre éstos y el presente proyecto. De esta forma se vio en qué se diferencia y destaca este proyecto de los ya existentes.

Tras este proceso de investigación, se procedió al diseño y al desarrollo del programa. El programa se realizó íntegramente en la plataforma LabVIEW [14] [15] [16] y se implementó en un transceptor NI USRP 2920 [7] – [10]. Sin embargo, a lo largo del desarrollo del programa surgieron varias complicaciones y varias alternativas de diseño, como se comentó en capítulo 6. Aun así, se logró realizar el proyecto con éxito.

Después, el proyecto pasó por un periodo de pruebas y evaluación cuyo fin era ver si cumplía con los objetivos marcados al principio del proyecto. Al hacer esto se vio que el proyecto implementaba correctamente las funcionalidades de una estación de tierra, que se comunicaba adecuadamente con el satélite, y que el programa era sencillo y fácil de utilizar. Por lo tanto se comprobó que el proyecto había cumplido con éxito los objetivos planteados al comienzo.

Consecuentemente, se puede afirmar que tanto los objetivos docentes como los objetivos técnicos del proyecto se han logrado muy satisfactoriamente, ya que se ha creado un simulador muy visual que servirá para explicar a los alumnos estos temas de una forma mucho más fácil y atrayente. Además, se ha conseguido la comunicación entre la estación de tierra y el satélite ajustándose a la normativa vigente.

Para finalizar, simplemente decir que el proyecto ha aportado mucho al alumno que lo ha realizado. En primer lugar, ha permitido hacer una gran ampliación de los conocimientos que poseía el alumno en los temas relacionados con satélites. En segundo lugar, también ha permitido al alumno aprender a utilizar herramientas que le eran totalmente desconocidas, como la plataforma LabVIEW o los transceptores NI USRP. Y en tercer lugar, el hecho de realizar este

proyecto de forma conjunta ha incrementado la capacidad del alumno de trabajar en equipo.

8.2. Futuras líneas de trabajo

Como se ha comentado a lo largo de la memoria, este proyecto se ha diseñado constantemente pensando en el futuro. Es un proyecto que está pensado para contribuir con la docencia de la Universidad, por lo que también está pensado para que las siguientes generaciones lo sigan desarrollando y mejorando.

Las futuras líneas de trabajo que podría tener este proyecto son las siguientes:

En primer lugar, se podrían implementar algunos parámetros o funcionalidades que no se han podido implementar en este proyecto principalmente por falta de tiempo pero que sí se han tenido en cuenta a la hora de realizarlo.

Un ejemplo de esto es el campo PEC (Packet Error Control) del paquete de telemetría (véase apartado 5.1). Este campo sirve para detectar posibles fallos en el paquete, permitiendo a la estación de tierra verificar la integridad del mensaje completo. La implementación de esta funcionalidad podría ser mediante una comprobación de *checksum*, un código CRC, etc.

Sin embargo, aunque esto no se ha implementado, sí que se ha tenido en cuenta a la hora de realizar el programa, ya que estos 16 bits sí que son leídos, pero no se realiza ninguna acción con ellos.

Además de esto, también se podrían implementar paquetes con otro formato distinto sin perder el actual, ya que como se comentó en el apartado 5.1, hay un campo en la cabecera del paquete llamado *Version Number* que indica el número de versión del paquete. A esta primera versión de los paquetes que se ha realizado en el proyecto se le ha asignado el número 0. Por lo tanto, si se quisieran añadir otras versiones, no habría que hacer más que cambiar el número de versión en los paquetes e implementar un sistema con el nuevo formato de paquetes. Es más, gracias el campo *Version Number* se podría conseguir incluso transmitir en una comunicación paquetes de varias versiones.

En segundo lugar, otra futura línea de trabajo sería implementar más servicios y funcionalidades de la estación de tierra. En el estándar de la ECSS número ECSS-E-70-41A [17] (que es en el que se basa el proyecto) vienen recogidos muchos más servicios que no han podido ser implementados por falta de tiempo.

Por otro lado, además de los servicios recogidos en el estándar [17], también cabe la posibilidad de implementar otros servicios específicos que no vienen incluidos en el estándar, al igual que se ha hecho con los servicios 32 y 64 de este

proyecto. Esto abre un abanico casi infinito de nuevas funciones y servicios que podrían ser implementados en la estación de tierra.

En tercer lugar, como ya se comentó en el apartado 6.1, otra posible mejora podría ser unir las funciones de telemetría y telecomandos de la estación de tierra en un único programa para que existiera comunicación entre ambas partes. Esto abriría la posibilidad de realizar un sistema de envío y recepción de paquetes con confirmación (ACK). Es decir, que cada vez que la estación de tierra recibiese un paquete del satélite le enviaría de vuelta un paquete de confirmación. Si pasado un tiempo el satélite no recibe este paquete, volvería a enviar de nuevo la información. Y lo mismo desde la estación de tierra.

Y en cuarto y último lugar, otra futura línea de trabajo podría ser ampliar el número de satélites o de estaciones de tierra simulados. Esta mejora también se ha tenido en cuenta desde el principio del proyecto. Por ese motivo se reservó un campo en la cabecera de los datos (Data Field Header) llamado *Destination ID* que se utiliza para indicar a quién va dirigido el paquete en el caso de que haya más de una estación de tierra o más de un satélite. Esta mejora sería fácil implementarla pero, al igual que los casos anteriores, no se ha podido llevar a cabo por falta de tiempo.

Como se puede observar, este proyecto no finaliza aquí, sino que tiene todavía un gran abanico de posibilidades abierto ante él y una infinidad de nuevas versiones y mejoras del proyecto. Queda por lo tanto un largo camino por delante que recorrer, y que recorrerán los futuros estudiantes de las generaciones venideras.

ANEXO A. Esquema de las tramas de telemetría y telecomandos

A continuación se muestran los formatos de los paquetes de telemetría y telecomandos que han sido utilizados tanto en esta parte del proyecto como en las partes realizadas por los otros dos miembros del equipo.

Paquete de Telemetría y Telecomandos

Packet Header (48 Bits)							Packet Data Field (Variable)		
Packet ID				Packet Sequence Control		Packet Length	Data Field Header	Source Data	Packet Error Control
Version Number (=0)	Type (=0)	Data Field Header Flag	Applica-tion Process ID	Grouping Flags	Source Sequence Count				
3	1	1	11	2	14				
16				16		16	32	Variable	16

Tabla A.1: Campos del paquete de Telemetría/Telecomandos [Figura 4[17]]

Campo *Data Field Header* del paquete de Telemetría

Data Field Header – Telemetría (32 Bits)					
Spare	TM Source Packet PUS Version Number	Spare	Service Type	Service Subtype	Destination ID
1 bit	3 bits	4 bits	8 bits	8 bits	8 bits

Tabla A.2: Cabecera del campo de datos de telemetría

Campo *Data Field Header* del paquete de Telecomandos

Data Field Header – Telecomandos (32 Bits)					
CCSDS Secondary Header Flag	TC Packet PUS Version Number	Ack	Service Type	Service Subtype	Source ID
1 bit	3 bits	4 bits	8 bits	8 bits	8 bits

Tabla A.3: Cabecera del campo de datos de telecomandos

ANEXO B. Resumen de Telecomandos y Telemetría

A continuación se muestra una tabla que contiene todos los servicios (tipo y subtipo) implementados en la simulación de la comunicación por satélite. Se incluyen, además de los servicios de telemetría, los servicios de telecomandos realizados por los otros miembros del equipo del proyecto. Juntos constituyen una lista completa de todos los servicios que realiza la estación de tierra y el satélite.

TM → Telemetría.
TC → Telecomando.

TC/TM	Tipo	Subtipo	Descripción del Servicio
			Servicio 1: Verificación de TC
TM	1	1	TC acceptance success report
TM	1	2	TC acceptance failure report
TM	1	7	TC execution success report
TM	1	8	TC execution failure report
			Servicio 3: Housekeeping y diagnóstico de datos
TC	3	1	Clear housekeeping parameter report generation
TC	3	3	Enable housekeeping report generation
TC	3	4	Disable housekeeping report generation
TC	3	5	Request housekeeping report generation period
TM	3	6	Housekeeping parameter report
TC	3	7	Update housekeeping report generation period
TC	3	8	Define housekeeping report interval
TM	3	9	Housekeeping report interval updated
			Servicio 5: Eventos
TM	5	1	Normal progress report
TM	5	2	Error / Anomaly report – Low severity – Warning
TM	5	3	Error / Anomaly report – Medium severity – Ground action
TM	5	4	Error / Anomaly report – High severity – On-board action
TC	5	5	Enable event report generation
TC	5	6	Disable event report generation
TC	5	16	Clear event log
TC	5	17	Report enabled event packets
TM	5	18	Enabled event packets report
TC	5	19	Report Disabled event packets
TM	5	20	Disabled event packets report

			Servicio 6: Gestión de memoria
TC	6	1	Load data into memory
TC	6	2	Dump memory area
TM	6	3	Memory dump
TC	6	4	Check memory
TM	6	5	Memory check report
TC	6	16	Preload ROM data
			Servicio 9: Gestión del tiempo
TC	9	1	Set OBT
TC	9	2	Time request
TM	9	3	Time management
			Servicio 11: Gestión del planificador de telecomandos
TC	11	1	Enable telecommands schedule
TC	11	2	Disable telecommands schedule
TC	11	3	Reset telecommands schedule
TC	11	4	Insert telecommand in command schedule
TC	11	5	Delete telecommand in command schedule
TC	11	6	Time shift to selected telecommand
TC	11	7	Report subset of command Schedule
TM	11	8	Detailed Schedule report
			Servicio 12: Monitorización de a bordo
TC	12	1	Enabling monitoring of parameters
TC	12	2	Disable monitoring of parameters
TC	12	3	Change máximum report delay
TC	12	4	Clear monitoring list
TC	12	7	Add parameter to monitoring list
TC	12	8	Delete parameter to monitoring list
TC	12	9	Report current monitoring list
TM	12	10	Current monitoring list report
			Servicio 15: Test de conexión
TC	15	1	Request connection test
TM	15	2	Connection test report
			Servicio 32: Instrumento sensor de temperatura
TC	32	1	Switch on instrument
TC	32	2	Switch off instrument
TC	32	3	Change time acquisition interval
TC	32	4	Request for time acquisition interval
TM	32	5	Time acquisition interval report
TC	32	6	Update time acquisition interval
TC	32	7	Request data
TM	32	8	Data from instrument
TC	32	16	Test instrument
TM	32	17	Instrument test report
TM	32	20	Transmission aborted

			Servicio 64: Instrumento de imágenes
TC	64	1	Switch on instrument
TC	64	2	Switch off instrument
TC	64	3	Change time acquisition interval
TC	64	4	Request for time acquisition interval
TM	64	5	Time acquisition interval report
TC	64	6	Update time acquisition interval
TC	64	7	Request data
TM	64	8	Data from instrument
TC	64	16	Test instrument
TM	64	17	Instrument test report
TM	64	20	Transmission aborted
			Servicio 128: Reboot
TC	128	1	Reboot

Tabla B.1: Servicios implementados

ANEXO C. Módulos auxiliares

En este anexo se explican los dos módulos auxiliares utilizados en este proyecto. Sin embargo, el hecho de que sean auxiliares no implica que no sean importantes, pues son los módulos que se usan con más frecuencia. Es más, todos los demás módulos principales los utilizan. El hecho de que se consideren auxiliares es simplemente porque no desempeñan las funcionalidades de ninguno de los servicios implementados. Estos dos módulos son los siguientes:

Binario a decimal (subVI).vi

La función de este módulo auxiliar es la de transformar arrays de bits en binario a números en decimal. Este módulo es constantemente utilizado en el programa, y su símbolo es:

Figura C.1: Símbolo de *Binario a decimal (subVI).vi*

Y en cuanto al funcionamiento del módulo, éste simplemente va leyendo los elementos del array binario, los eleva a la potencia de 2 correspondiente a la posición que ocupan en el array, y los suma. El diagrama de flujo se puede ver a continuación en la Figura C.2.

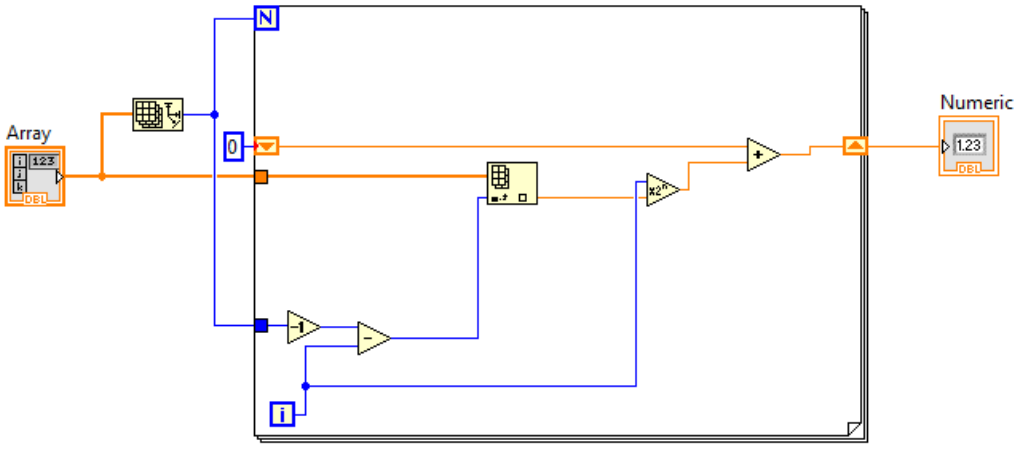


Figura C.2: *Binario a decimal (subVI).vi*

Escribir log (subVI).vi

Este otro módulo auxiliar es el encargado de guardar los datos en el registro (*log*). Este módulo se utiliza cada vez que se obtienen datos de cualquiera de los servicios implementados. El símbolo de este módulo es el siguiente:

Figura C.3: Símbolo de *Escribir log (subVI).vi*

En cuanto a la implementación, el módulo lo primero que hace es comprobar si en el directorio del programa existe ya un fichero llamado *log.txt*. Si es así, se abrirá dicho fichero para continuar escribiendo en él; y si no, se creará el fichero. Una vez creado o abierto el fichero, se busca la última posición en la que hay datos y a partir de ahí se escribe en él la fecha y hora y la cadena de texto que le llega como entrada. Una vez escrito, el fichero se cierra de nuevo. El diagrama de bloques del módulo se puede ver a continuación en la Figura C.4.

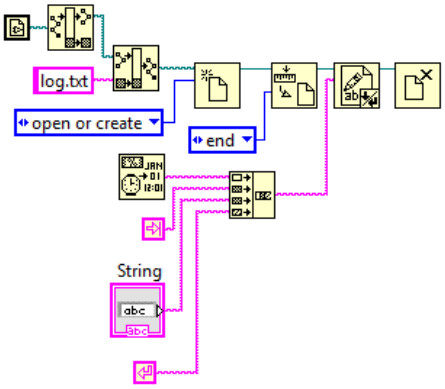
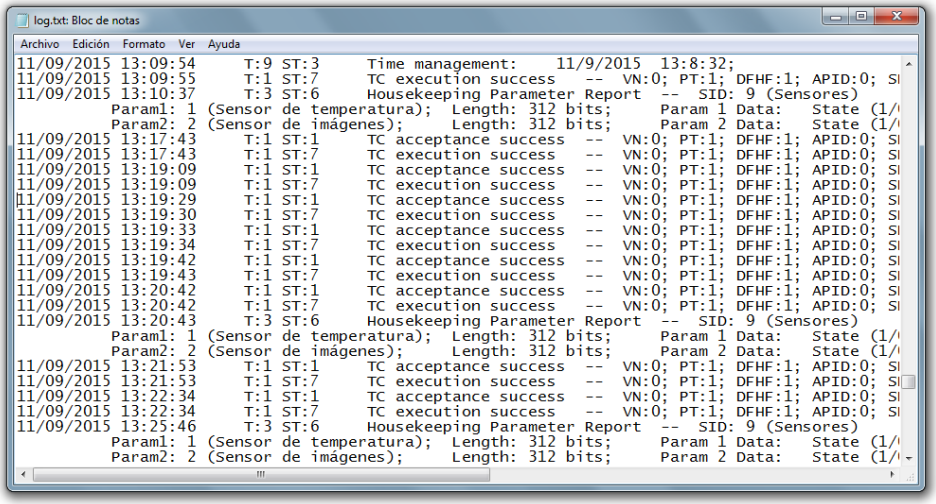


Figura C.4: *Escribir log (subVI).vi*

Un ejemplo del registro *log.txt* en el que guarda los datos se puede ver a continuación en la Figura C.5.


log.txt: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
11/09/2015 13:09:54     T:9 ST:3     Time management:    11/9/2015  13:8:32;
11/09/2015 13:09:55     T:1 ST:7     TC execution success   --  VN:0; PT:1; DFHF:1; APID:0; SI
11/09/2015 13:10:37     T:3 ST:6     Housekeeping Parameter Report  --  SID: 9 (Sensores)
          Param1: 1 (Sensor de temperatura);  Length: 312 bits;     Param 1 Data:   State (1/
          Param2: 2 (Sensor de imágenes);     Length: 312 bits;     Param 2 Data:   State (1/
11/09/2015 13:17:43     T:1 ST:1     TC acceptance success  --  VN:0; PT:1; DFHF:1; APID:0; SI
11/09/2015 13:17:43     T:1 ST:7     TC execution success   --  VN:0; PT:1; DFHF:1; APID:0; SI
11/09/2015 13:19:09     T:1 ST:1     TC acceptance success  --  VN:0; PT:1; DFHF:1; APID:0; SI
11/09/2015 13:19:09     T:1 ST:7     TC execution success   --  VN:0; PT:1; DFHF:1; APID:0; SI
11/09/2015 13:19:29     T:1 ST:1     TC acceptance success  --  VN:0; PT:1; DFHF:1; APID:0; SI
11/09/2015 13:19:30     T:1 ST:7     TC execution success   --  VN:0; PT:1; DFHF:1; APID:0; SI
11/09/2015 13:19:33     T:1 ST:1     TC acceptance success  --  VN:0; PT:1; DFHF:1; APID:0; SI
11/09/2015 13:19:34     T:1 ST:7     TC execution success   --  VN:0; PT:1; DFHF:1; APID:0; SI
11/09/2015 13:19:42     T:1 ST:1     TC acceptance success  --  VN:0; PT:1; DFHF:1; APID:0; SI
11/09/2015 13:19:43     T:1 ST:7     TC execution success   --  VN:0; PT:1; DFHF:1; APID:0; SI
11/09/2015 13:20:42     T:1 ST:1     TC acceptance success  --  VN:0; PT:1; DFHF:1; APID:0; SI
11/09/2015 13:20:42     T:1 ST:7     TC execution success   --  VN:0; PT:1; DFHF:1; APID:0; SI
11/09/2015 13:20:43     T:3 ST:6     Housekeeping Parameter Report  --  SID: 9 (Sensores)
          Param1: 1 (Sensor de temperatura);  Length: 312 bits;     Param 1 Data:   State (1/
          Param2: 2 (Sensor de imágenes);     Length: 312 bits;     Param 2 Data:   State (1/
11/09/2015 13:21:53     T:1 ST:1     TC acceptance success  --  VN:0; PT:1; DFHF:1; APID:0; SI
11/09/2015 13:21:53     T:1 ST:7     TC execution success   --  VN:0; PT:1; DFHF:1; APID:0; SI
11/09/2015 13:22:34     T:1 ST:1     TC acceptance success  --  VN:0; PT:1; DFHF:1; APID:0; SI
11/09/2015 13:22:34     T:1 ST:7     TC execution success   --  VN:0; PT:1; DFHF:1; APID:0; SI
11/09/2015 13:25:46     T:3 ST:6     Housekeeping Parameter Report  --  SID: 9 (Sensores)
          Param1: 1 (Sensor de temperatura);  Length: 312 bits;     Param 1 Data:   State (1/
          Param2: 2 (Sensor de imágenes);     Length: 312 bits;     Param 2 Data:   State (1/
```

Figura C.5: Ejemplo del registro *log.txt*

ANEXO D. Codificación de imágenes

En este anexo se explica brevemente cómo se codifica una imagen, ya que esto es necesario para llevar a cabo el servicio (64,8) encargado de transformar los datos enviados por el satélite a una imagen.

Toda imagen está formada por píxeles [40] (en este caso para la simulación del satélite se han escogido imágenes de 200x200 píxeles). Cada píxel es de un color, y este color se forma mediante la mezcla de los tres colores primarios: rojo (R), verde (G) y azul (B). Esto se ve de forma más sencilla en la siguiente figura:

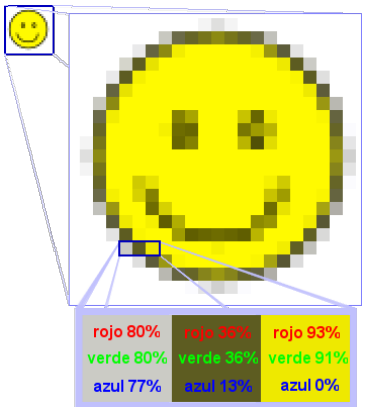


Figura D.1: Codificación de una imagen [40]

El nivel de intensidad de cada componente rojo, verde y azul se codifica con 8 bits, pudiendo alcanzar valores desde 0 hasta 255. Como se puede ver en la Figura D.1, cada píxel contiene la información de estas tres componentes. Por lo tanto, para codificar un píxel harán falta 24 bits. Y para codificar una imagen entera de 200x200 píxeles harán falta 960.000 bits.

ANEXO E. Manual de usuario

A continuación se presenta en este anexo un manual del programa implementado para que le sea más fácil al usuario utilizarlo y para aclarar cualquier posible duda acerca de su funcionamiento.

Cuando se ejecuta el módulo principal del programa (*Estación de tierra.vi*) aparece la siguiente pantalla de inicio:

Figura E.1: Pantalla de inicio

En ella se muestra el nombre del programa y se da la bienvenida al usuario.

Después se muestra el panel frontal con el que el usuario va a trabajar. Este panel frontal está formado por 10 pestañas. Cada una de ellas tiene una función en el programa y es lo que se va a explicar a continuación.

Pestaña *Info*

Esta primera pestaña es la principal y más importante del programa. En ella se mostrará constantemente al usuario toda la información de lo que está sucediendo en el programa. Esta pestaña se puede ver en la Figura E.2.

Cuenta con un panel de texto en el que constantemente se muestran los mensajes enviados desde el satélite. También cuenta con seis indicadores luminosos: dos de ellos (*Successful* y *Failure*) muestran al usuario si se ha producido o no algún fallo en el satélite, y los otros cuatro (*Aceptación Correcta, Ejecución Correcta, Aceptación Incorrecta* y *Ejecución incorrecta*) son utilizados únicamente por el servicio 1 de verificación de telecomandos. A la izquierda de la pantalla también se muestran dos indicadores numéricos con el tipo y subtipo del último paquete recibido por la estación de tierra. Y a la derecha de la pantalla hay

un botón (*Limpiar pantalla*) que sirve para volver a dejar la pantalla vacía. Sin embargo, borrar la pantalla no implica que se borren los datos guardados en el registro (*log*) del programa.

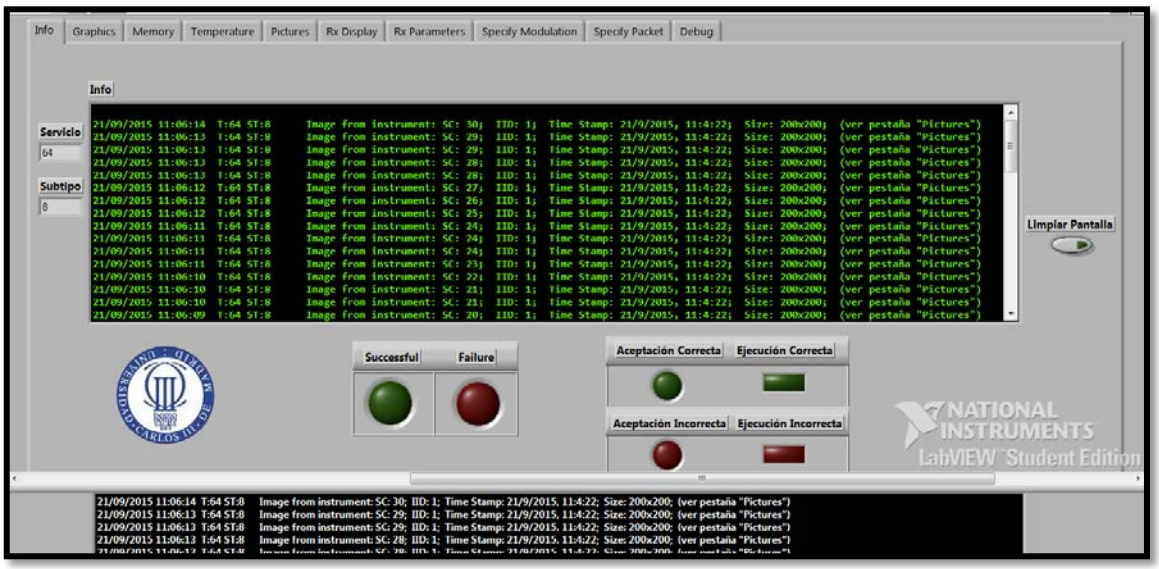


Figura E.2: Pestaña *Info*

Pestaña *Graphics*

Esta pestaña se utiliza únicamente para el servicio 3 de housekeeping y diagnóstico de datos. En ella se muestran de forma gráfica los datos enviados desde el satélite relativos a los parámetros que están siendo monitorizados. La pestaña consta de cuatro gráficas y cada una de ellas representa uno de estos parámetros. Éstos son: el estado del sensor de imágenes, el estado del sensor de temperatura, los datos de la temperatura del microprocesador y los datos de la frecuencia de funcionamiento del microprocesador. Véase la Figura E.3.

Figura E.3: Pestaña *Graphics*

Pestaña *Memory*

Esta pestaña se usa exclusivamente en el servicio 6 de gestión de memoria, concretamente en el comando (6,3). En ella se muestra el bloque de memoria que se ha enviado desde el satélite, como se puede ver en la Figura E.4.

Figura E.4: Pestaña *Memory*

Pestaña *Temperature*

En esta pestaña se muestra toda la información relativa a los datos de temperatura enviados por el satélite. Se puede ver a continuación en la Figura E.5.

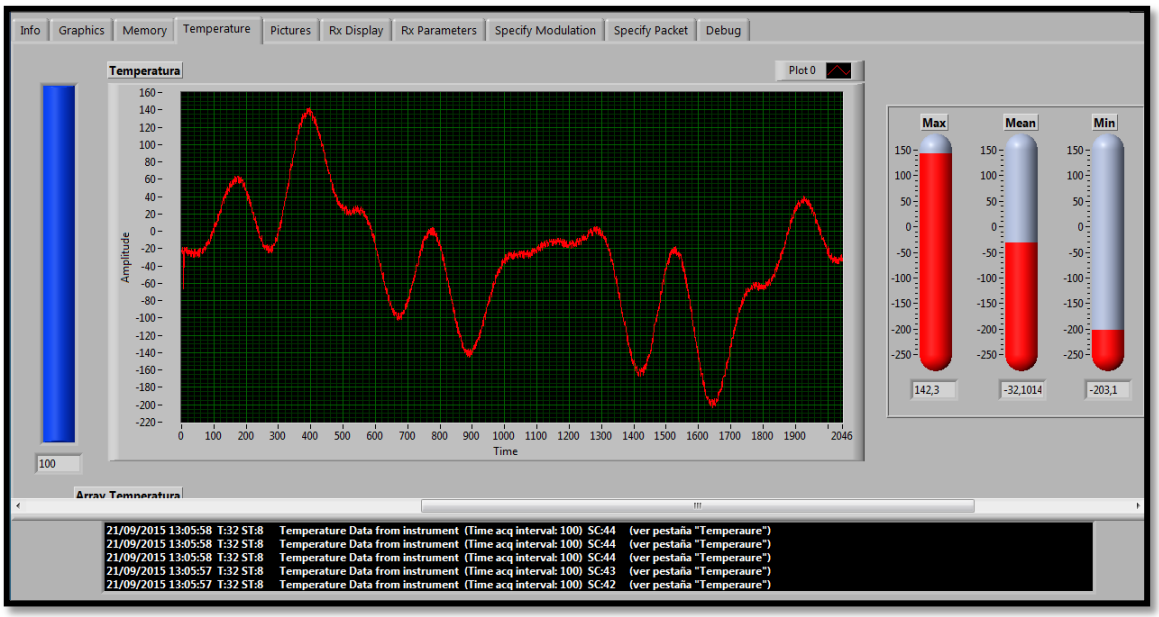


Figura E.5: Pestaña *Temperature*

Como se puede observar en la figura anterior, esta pestaña cuenta con una gráfica en el centro donde se representan los datos de temperatura recogidos y

enviados por el satélite. A la izquierda se muestra una barra de progreso que indica el porcentaje de datos recibidos en función de los totales. Y a la derecha hay tres termómetros que indican las temperaturas máxima, media y mínima de los datos, respectivamente.

Pestaña *Pictures*

Esta pestaña está relacionada con el servicio 64. Concretamente con el comando (64,8) que es el encargado de recibir los datos enviados por el satélite y reconstruir la imagen original. A continuación se muestra esta pestaña:

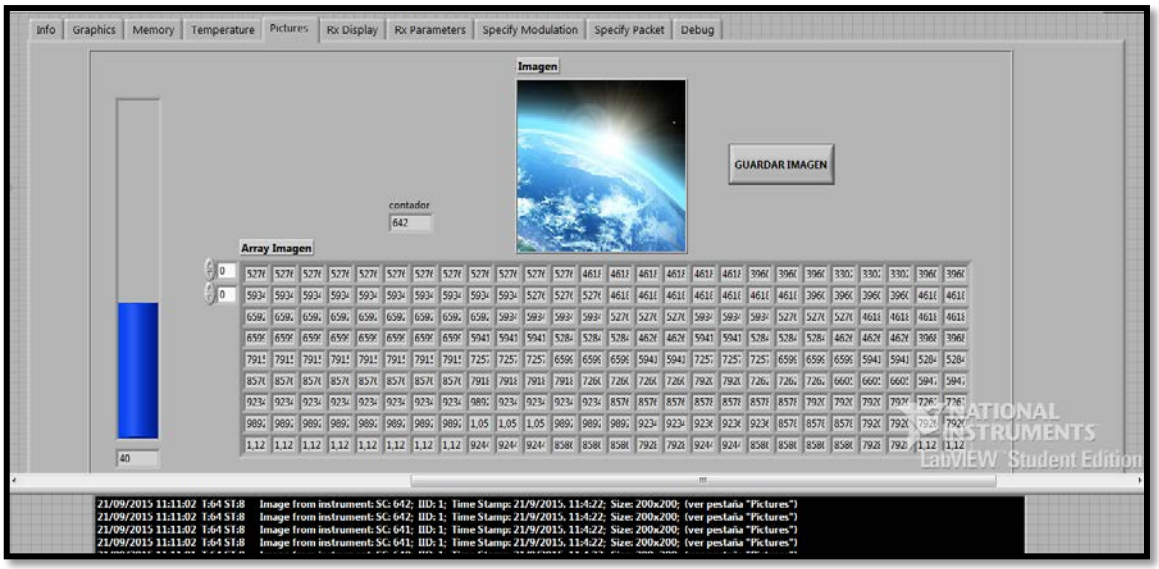


Figura E.6: Pestaña *Pictures*

En esta pestaña se puede ver en el centro la imagen enviada desde el satélite. Al igual que en la anterior pestaña, en ésta también hay una barra de progreso que muestra el porcentaje de paquetes recibidos. También hay un indicador numérico llamado *contador* que muestra el número de paquetes de la imagen que se han recibido. Debajo de la imagen se muestra una porción del array que contiene los valores de los píxeles de la imagen. Y a la derecha hay un botón que permite guardar la imagen en el ordenador, como se puede ver en la Figura E.7.

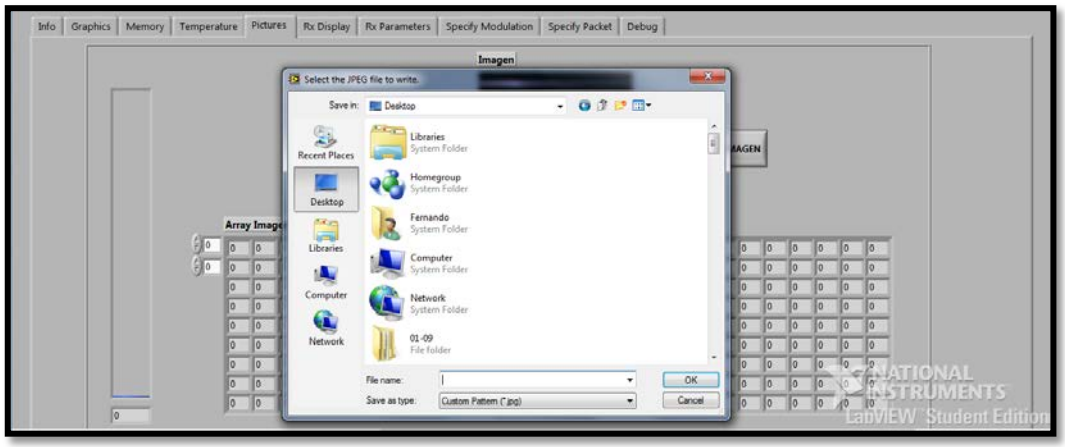


Figura E.7: Ejecución del botón *GUARDAR IMAGEN*

Pestaña *Rx Display*

Esta pestaña sirve para ver la señal recibida por la estación de tierra. Arriba a la derecha se puede ver la amplitud de la señal recibida. Abajo a la derecha se encuentra una gráfica que representa la constelación de la señal recibida. Y a la izquierda se puede ver el array de bits que contiene esa señal. Esto se puede ver a continuación en la Figura E.8.

Figura E.8: Pestaña *Rx Display*

Pestaña *Rx Parameters*

Esta pestaña sirve para configurar los parámetros de recepción del transceptor NI USRP. La pestaña se muestra a continuación.

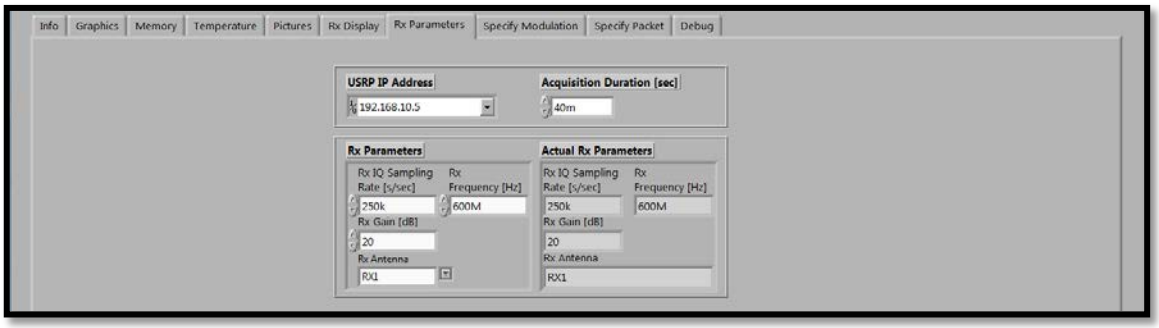


Figura E.9: Pestaña *Rx Parameters*

Como se puede ver en la Figura E.9, en la parte superior de la pestaña se puede configurar la dirección IP del transceptor y el tiempo de adquisición del mismo. Debajo, se pueden configurar también la tasa de muestreo, la frecuencia de recepción, la ganancia de recepción y el puerto por el que se va a recibir. Se recomienda que, salvo la dirección IP, se utilicen los mismos valores que aparecen en la Figura E.9.

Pestaña *Specify Modulation*

Esta pestaña sirve para elegir el tipo de modulación que se quiere utilizar y para configurar los parámetros del filtro. Se puede ver a continuación.

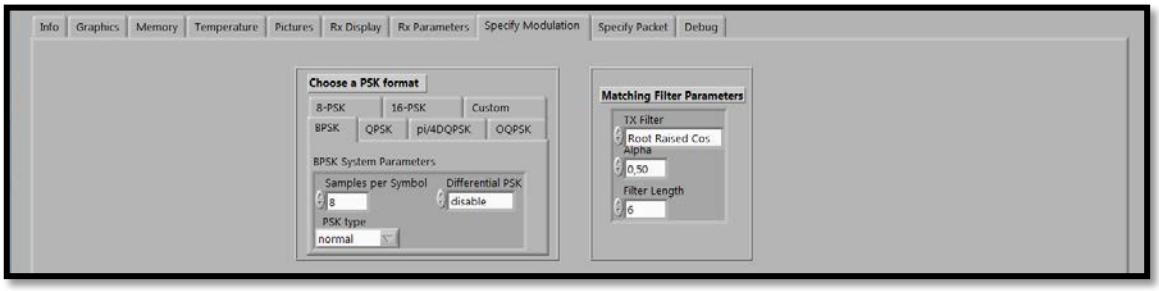


Figura E.10: Pestaña *Specify Modulation*

A la izquierda se puede elegir la modulación que se desea utilizar, y a la derecha se pueden configurar los distintos parámetros del filtro. Se recomienda utilizar la modulación BPSK ya que es la más simple y la que mejor funciona para este proyecto. El resto de parámetros se recomienda que se ajusten a los valores mostrados en la Figura E.10.

Pestaña *Specify Packet*

Esta pestaña permite configurar todos los parámetros relativos a los paquetes. En primer lugar, el número de bits de guarda, que se añaden al comienzo del paquete para evitar que se pierdan los primeros bits. En segundo lugar, el número de bits de sincronización, que permiten que el transceptor se sincronice. Y en tercer lugar, el número de bits por mensaje. Debajo de esto también se puede configurar el orden de la secuencia de sincronización.

Se recomienda utilizar los valores que se muestran a continuación en la Figura E.11.

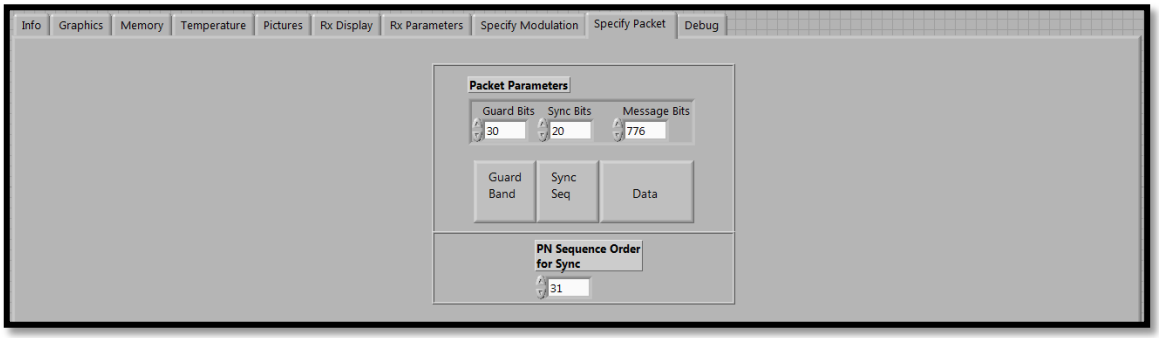


Figura E.11: Pestaña *Specify Packet*

Pestaña *Debug*

En esta pestaña se puede ver la relación señal a ruido de la señal recibida. Consta de una gráfica a la izquierda en la que se pueden ver representadas la potencia de la señal recibida (color blanco en la gráfica) y la potencia del ruido (color rojo en la gráfica). A la derecha de la gráfica se puede ver el valor de la

última amplitud de la señal y del ruido medida por el transceptor. Y debajo se puede ver un indicador que se activa cuando se produce algún error en el transceptor NI USRP. La pestaña se puede ver a continuación.

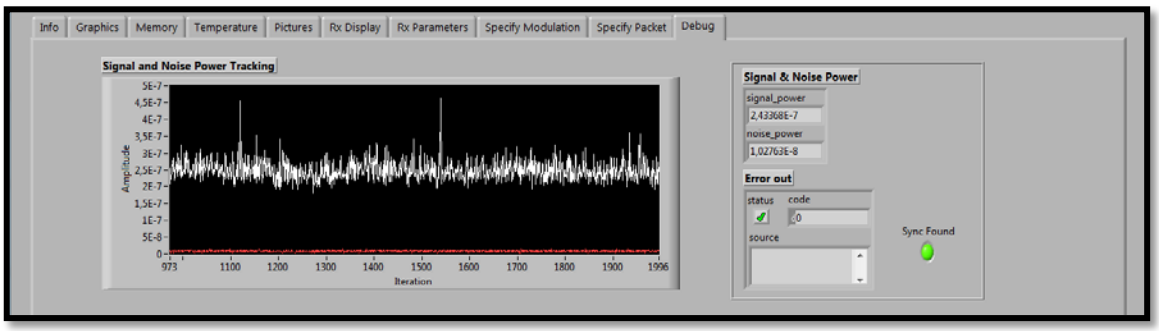


Figura E.12: Pestaña *Debug*

Común a todas estas pestañas, se muestra también una pequeña pantalla de texto en la parte de abajo del panel frontal. Ésta muestra las últimas cinco líneas aparecidas en la pantalla principal (*Info*).

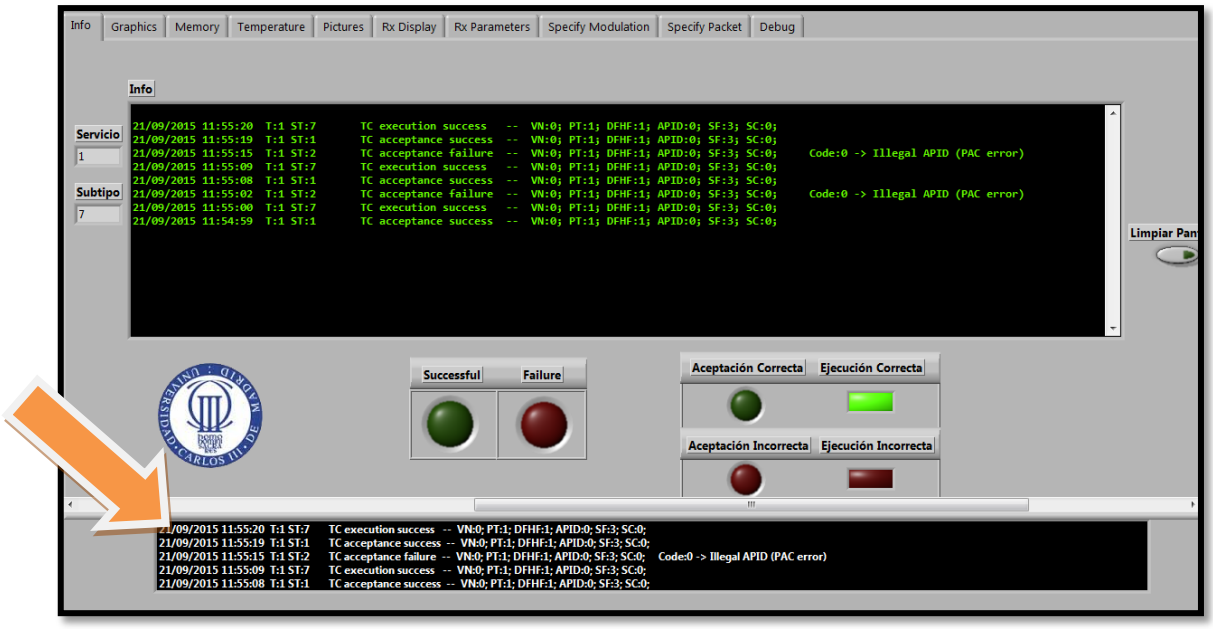


Figura E.13: Ejemplo de la pantalla inferior

Esta pantalla siempre permanece visible en cualquiera de las pestañas anteriores. Se hizo con el objetivo de que el usuario pudiera ver siempre la información recibida independientemente de la pestaña en la que se encontrara.

Bibliografía

[1] Maini, A.K. & Agrawal, V. 2014, *Satellite technology: principles and applications*, Wiley, Chichester, West Sussex.

[2] Daniel Rodríguez Mogollón, "Simulador de EGSE (Electric Ground Support Equipment) [TELECOMANDOS] mediante plataforma Software Defined Radio", 2015.

[3] Fernando García Arias, "Simulador simple de OBDH (On Board Data Handling) basado en Software Defined Radio", 2015.

[4] Tuttlebee, W.H.W. & ebrary, I. 2002; 2003, *Software defined radio: origins, drivers and international perspectives*, 1st edn, J. Wiley, New York.

[5] Tuttlebee, W.H.W. & ebrary, I. 2002; 2003*, Software defined radio: enabling technologies*, 1st edn, J. Wiley, New York.

[6] Wireless Innovation Forum, "What is Software Defined Radio?" [En línea]. Disponible en: http://www.wirelessinnovation.org/index.php?option=com_content&view=article&id=134:What_is_SDR&catid=19:site-content&Itemid=76. [Último acceso: 7 sep. 2015].

[7] National Instruments, "What Is NI USRP Hardware?" [En línea]. Disponible en: http://www.ni.com/white-paper/12985/en/. [Último acceso: 10 sep. 2015].

[8] National Instruments, "USRP-2920" [En línea]. Disponible en: http://sine.ni.com/nips/cds/view/p/lang/es/nid/209948. [Último acceso: 10 sep. 2015].

[9] National Instruments, "NI USRP-2920 Device Specifications" [En línea]. Disponible en: http://www.ni.com/pdf/manuals/375839a.pdf. [Último acceso: 10 sep. 2015].

[10] National Instruments, "NI USRP-2920 Datasheet" [En línea]. Disponible en: http://www.ni.com/datasheet/pdf/en/ds-355#detailedspecs-section. [Último acceso: 10 sep. 2015].

[11] Ministerio de Industria, Energía y Turismo, "Cuadro Nacional de Atribución de Frecuencias (CNAF)" [En línea]. Disponible en: http://www.minetur.gob.es/telecomunicaciones/Espectro/Paginas/CNAF.aspx. [Último acceso: 4 sep. 2015].

[12] Wikipedia, “Cuadro Nacional de Atribución de Frecuencias” [En línea]. Disponible en: https://es.wikipedia.org/wiki/Cuadro_Nacional_de_Atribuci%C3%B3n_de_Frecuencias. [Último acceso: 4 sep. 2015].

[13] NI, “National Instruments” [En línea]. Disponible en: http://spain.ni.com/. [Último acceso: 10 sep. 2015].

[14] National Instruments, “LabVIEW” [En línea]. Disponible en: http://www.ni.com/labview/esa/. [Último acceso: 10 sep. 2015].

[15] Lajara Vizcaíno, J.R. & Pelegrí Sebastiá, J. 2011, *LabVIEW: entorno gráfico de programación*, 2ª edn, Marcombo, Barcelona.

[16] National Instruments, “Fundamentos del entorno de LabVIEW” [En línea]. Disponible en: http://www.ni.com/getting-started/labview-basics/esa/environment. [Último acceso: 10 sep. 2015].

[17] European Cooperation for Space Standardization, “Space Engineering: Ground systems and operations – Telemetry and telecommand packet utilization,” ECSS-E-70-41A, January 2003.

[18] NASA, “New Horizons” [En línea]. Disponible en: https://www.nasa.gov/mission_pages/newhorizons/main/index.html. [Último acceso: 2 sep. 2015].

[19] Quo, “¿Cuántos artefactos están orbitando la Tierra a la vez?” [En línea]. Disponible en: http://www.quo.es/tecnologia/cuantos-artefactos-estan-orbitando-la-tierra-a-la-vez. [Último acceso: 2 sep. 2015].

[20] IEEE, “Institute of Electrical and Electronics Engineers” [En línea]. Disponible en: https://www.ieee.org/. [Último acceso: 16 sep. 2015].

[21] Terma, “Simuladores de satélite” [En línea]. Disponible en: http://www.terma.com/space/ground-segment/satellite-simulators/. [Último acceso: 14 sep. 2015].

[22] Terma, “Simulador SIMSAT” [En línea]. Disponible en: http://www.terma.com/media/148537/simsat_simulators.pdf. [Último acceso: 14 sep. 2015].

[23] Schor, D.; Kinsner, W.; Thoren, A., "Satellite ground station emulator: An architecture and implementation proposal," in Electrical and Computer Engineering, 2009. CCECE '09. Canadian Conference, pp.868-873, 3-6 May 2009.

[24] Fischer, M.; Scholtz, A.L., "Design of a Multi-mission Satellite Ground Station for Education and Research," in Advances in Satellite and Space Communications (SPACOMM), 2010 Second International Conference, pp.58-63, 13-19 June 2010.

[25] Wikipedia, “Global Educational Network for Satellite Operations (GENSO)” [En línea]. Disponible en: https://en.wikipedia.org/wiki/Global_Educational_Network_for_Satellite_Operations. [Último acceso: 16 sep. 2015].

[26] BRITE-Constellation, “Bright Target Explorer” [En línea]. Disponible en: http://www.brite-constellation.at/. [Último acceso: 16 sep. 2015].

[27] BOE, “Real Decreto 1066/2001” [En línea]. Disponible en: https://www.boe.es/boe/dias/2001/09/29/pdfs/A36217-36227.pdf. [Último acceso: 8 sep. 2015].

[28] ITU, “Sir Arthur C. Clarke — Visionario de la era espacial” [En línea]. Disponible en: https://www.itu.int/itunews/manager/display.asp?lang=es&year=2008&issue=03&ipage=Arthur-Clarke&ext=html. [Último acceso: 4 sep. 2015].

[29] NASA, “Sputnik” [En línea]. Disponible en: http://history.nasa.gov/sputnik/. [Último acceso: 4 sep. 2015].

[30] Russian Space Web, “Sputnik 2” [En línea]. Disponible en: http://www.russianspaceweb.com/sputnik2.html. [Último acceso: 4 sep. 2015].

[31] Russian Space Web, “Sputnik 3” [En línea]. Disponible en: http://www.russianspaceweb.com/sputnik3.html. [Último acceso: 4 sep. 2015].

[32] NASA, “Explorer-1 overview” [En línea]. Disponible en: https://www.nasa.gov/mission_pages/explorer/explorer-overview.html. [Último acceso: 4 sep. 2015].

[33] Wikipedia, “Vanguard-1” [En línea]. Disponible en: https://es.wikipedia.org/wiki/Vanguard_1. [Último acceso: 4 sep. 2015].

[34] Mobile Experts, “Homepage” [En línea]. Disponible en: http://www.mobile-experts.net/. [Último acceso: 7 sep. 2015].

[35] Wireless Innovation Forum, “Software Defined Radio - Rate of Adoption” [En línea]. Disponible en: http://www.wirelessinnovation.org/SDR_Rate_of_Adoption. [Último acceso: 7 sep. 2015].

[36] Wikipedia, “Gigabit Ethernet” [En línea]. Disponible en: https://en.wikipedia.org/wiki/Gigabit_Ethernet. [Último acceso: 10 sep. 2015].

[37] Wikipedia, “VHDL” [En línea]. Disponible en: https://es.wikipedia.org/wiki/VHDL. [Último acceso: 10 sep. 2015].

[38] National Instruments Comunity, “Documents” [En línea]. Disponible en: https://decibel.ni.com/content/docs/DOC-18801. [Último acceso: 17 feb. 2015].

[39] Agencia Tributaria, “Estimación directa simplificada” [En línea]. Disponible en: http://www.agenciatributaria.es/AEAT.internet/Inicio/_Segmentos_/Empresas_y_profesionales/Empresarios_individuales_y_profesionales/Rendimientos_de_actividades_economicas_en_el_IRPF/Regimenes_para_determinar_el_rendimiento_de_las_actividades_economicas/Estimacion_Directa_Simplificada.shtml. [Último acceso: 15 sep. 2015]

[40] Wikipedia, “Imagen de mapa de bits” [En línea]. Disponible en: https://es.wikipedia.org/wiki/Imagen_de_mapa_de_bits. [Último acceso: 22 sep. 2015].